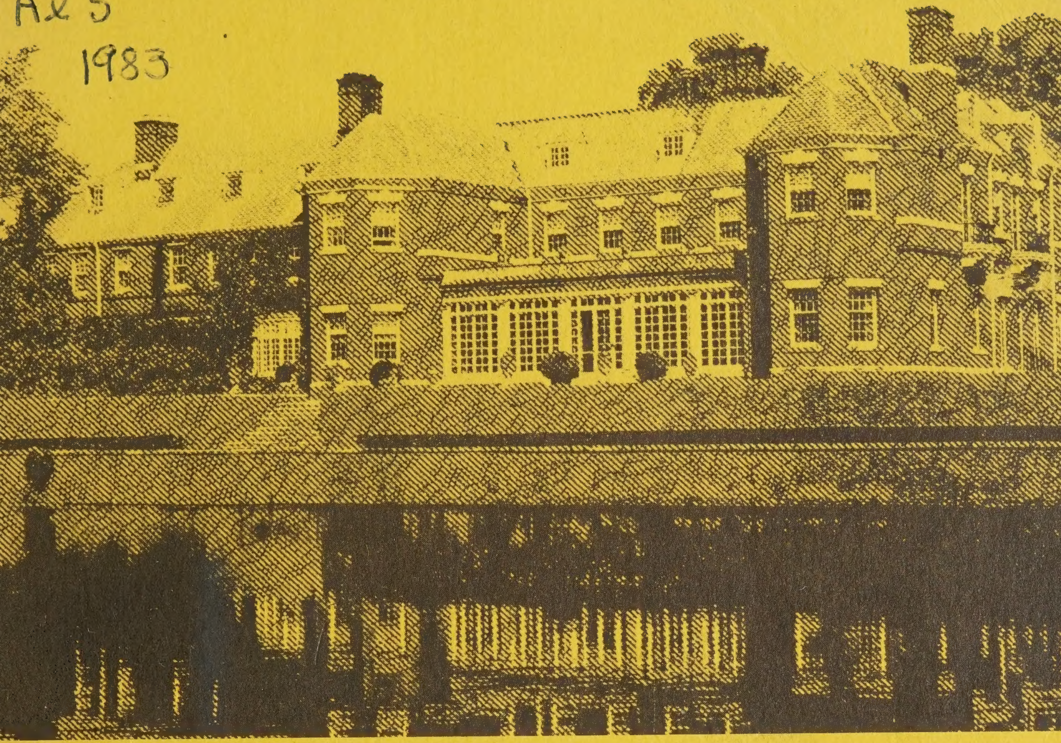
621.3811
Al5
1983



Proceedings

TWENTY-FIRST ANNUAL ALLERTON CONFERENCE ON COMMUNICATION, CONTROL, AND COMPUTING

October 5-7, 1983

Allerton House, Monticello, Illinois
Sponsored by the
Department of Electrical Engineering and the
Coordinated Science Laboratory of the
University of Illinois at Urbana-Champaign

PROCEEDINGS

TWENTY-FIRST ANNUAL ALLERTON CONFERENCE ON COMMUNICATION, CONTROL, AND COMPUTING

Tamer Başar
Donna J. Brown
H. Vincent Poor
Conference Co-Chairmen

Conference held
October 5-7, 1983
Allerton House
Monticello, Illinois

Sponsored by
The Department of Electrical Engineering
and
The Coordinated Science Laboratory
of the
UNIVERSITY OF ILLINOIS
at
Urbana-Champaign

Digitized by the Internet Archive
in 2026 with funding from
University of Illinois Urbana-Champaign

https://archive.org/details/proceedings1983alle_0

FOREWORD

The Twenty-First Annual Allerton Conference on Communication, Control, and Computing was held at Allerton House, Monticello, Illinois on October 5, 6, and 7, 1983. The conference was sponsored by the Department of Electrical Engineering and the Coordinated Science Laboratory of the University of Illinois at Urbana-Champaign. These PROCEEDINGS contain the papers presented at the Conference.

In addition to the large number of contributed papers, there were two organized sessions of invited papers. We are grateful to Professor Edward van der Meulen of the Katholieke Universiteit, Leuven, Belgium, for organizing and chairing a session on Multi-User Information Theory, and to Professor S. P. Bingulac of the University of Belgrade, Yugoslavia, for organizing and chairing a session on Computer Aided Design of Control Systems. Their time and efforts on behalf of the Conference are deeply appreciated. We would also like to express our gratitude to the other session chairmen and authors for their contributions to the success of the Conference. In addition, we thank The Square Knot Quartet of Urbana, for their musical contribution to the social program.

The support and encouragement of Professor G. W. Swenson, Jr., Head of the Department of Electrical Engineering, and Professor R. T. Chien, Director of the Coordinated Science Laboratory are appreciated and acknowledged. We would also like to thank our colleagues, Professors J. A. Abraham, T. Ü. Başar, J. B. Cruz, Jr., B. E. Hajek, I. N. Hajj, T. S. Huang, W. K. Jenkins, P. V. Kokotovic, M. C. Loui, D. E. Muller, D. C. Munson, Jr., J. H. Patel, W. R. Perkins, F. P. Preparata, M. B. Pursley, V. Ramachandran, D. V. Sarwate, T. N. Trick, and M. E. Van Valkenburg for their generous assistance in organizing the Conference.

We are grateful to Mr. Steven Williams for his handling of the business affairs of the Conference, to Ms. Rose Harris for her handling of registration both prior to and during the Conference, and to Mrs. Phyllis Young for her assistance with registration at the Conference. Finally, very special thanks are due to Mrs. Dixie Murphy for the gracious and impeccable assistance in all aspects of the organization of the Conference, from the initial preparation of the Conference announcement through Conference registration to the final editing of these PROCEEDINGS.

Tamer Başar

Donna J. Brown

H. Vincent Poor

TABLE OF CONTENTS

FOREWORD.. iii

I-A: MATHEMATICAL SYSTEM THEORY

Chairman: David L. Elliott
Washington University

A NOTE ON STATE TRAJECTORIES OF CAUSAL INPUT-OUTPUT SYSTEMS.. 1
W. L. Root

A POLYNOMIAL FRACTIONAL APPROACH TO LINEAR TIME-VARYING SYSTEMS.. 11
E. Emre

COEFFICIENT ASSIGNMENT FOR SISO TIME DELAY SYSTEMS.... 21
J. Chiasson

EXISTENCE OF DIAGONAL CONTROLLERS FOR A CLASS OF LINEAR TIME-INVARIANT SYSTEMS........................ 23
K. Natarajan and R. Balasubramanian

MODULE CONSIDERATIONS FOR FEEDBACK SYNTHESIS OF SENSITIVITY COMPARISONS............................ 25
K. P. Dudek, M. K. Sain and B. F. Wyman

SWITCHED LINEAR SYSTEMS: STABILITY AND OBSERVATION DESIGN.. 35
M. McIntyre and M. Vidyasagar

ON FEEDBACK EQUIVALENCE OF RETARDED AND NEUTRAL DELAY-DIFFERENTIAL EQUATIONS......................... 43
M. W. Spong

I-B: COMBINATORIAL ALGORITHMS

Chairman: Clyde P. Kruskal
University of Illinois

AN EXPERIMENTAL STUDY OF BIN PACKING................. 51
J. L. Bentley, D. S. Johnson, T. Leighton and C. C. McGeoch

VARIABLE SIZED BIN PACKING........................... 61
D. K. Friesen and M. A. Langston

LINEAR TIME UNIVERSAL COMPRESSION TECHNIQUES FOR DITHERED IMAGES BASED ON PATTERN MATCHING............ 70
A. Apostolico and E. Guerrieri

MULTI-RESOURCE EFFICIENT ALGORITHMS FOR INTEGER MULTIPLICATION....................................... 80
D. A. Carlson

621.3811
Al5
1983

Engin

GENERATION OF BINARY TREES OF BOUNDED HEIGHT.......... 90
C. C. Lee, D. T. Lee and C. K. Wong

A LINEAR CHARACTERIZATION OF NP-COMPLETE PROBLEMS..... 100
S. Ursic

I-C: DIGITAL IMAGE PROCESSING

Chairman: Thomas S. Huang
University of Illinois

TIME-SEQUENTIAL SAMPLING IN N-DIMENSIONAL EUCLIDEAN SPACE.. 110
J. P. Allenbach

SOME NEW RESULTS ON TREE CODING OF IMAGES............ 120
N. Moayeri, M. Naraghi-Pour and D. L. Neuhoff

RESTORATION OF DISPERSED IMPULSE SEQUENCES WITH AN EFFICIENT MAP ALGORITHM............................. 130
J. W. Sennott

ADAPTIVE GRAY SCALE MAPPING TO REMOVE REGISTRATION NOISE IN DIFFERENCE PICTURES......................... 132
T. F. Knoll and E. J. Delp

QUANTITATIVE ANALYSIS OF A MOMENT BASED EDGE OPERATOR... 142
P. H. Eichel and E. J. Delp

A STATISTICAL MODEL BASED APPROACH TO EDGE DETECTION IN TEXTURES... 152
R. L. Kashyap and R. D. Bauer

I-D: COMMUNICATION NETWORKS

Chairman: Bruce Hajek
University of Illinois

STABILITY AND THROUGHPUT IN VIRTUAL TIME CSMA......... 160
J. S. Meditch and C.-T. A. Lea

OPTIMAL FLOW CONTROL IN A CSMA/CD ENVIRONMENT......... 170
E. J. Coyle and A. A. Lazar

ON THE APPLICATION OF LINEAR PROGRAMMING TO OPTIMAL FLOW CONTROL... 180
A. A. Lazar and T. G. Robertazzi

THE THROUGHPUT TIME DELAY TRADE-OFF OF A BIRTH-DEATH PROCESS.. 190
H. K. Ahmad and A. A. Lazar

FLOW CONTROL IN ISDN BASED ON THE ESTIMATION OF THE VOICE TRAFFIC LOAD.................................. 200
F. Vakil and A. A. Lazar

AGGREGATION AND DECOMPOSITION IN THE PLANNING OF EVOLVING TELECOMMUNICATIONS NETWORKS.................. 202
A. D. Kasavin and J. R. Myers

II-A: GRAPHS AND VLSI

Chairman: Vijaya Ramachandran
University of Illinois

A MIXED-INTEGER LINEAR PROGRAMMING PROBLEM WHICH IS EFFICIENTLY SOLVABLE................................. 204
C. E. Leiserson and J. B. Saxe

EDGE GRAMMARS AND PARALLEL COMPUTATION................ 214
F. Berman

GRAPH SEPARATION AND SEARCH NUMBER................... 224
J. A. Ellis, I. H. Sudborough and J. S. Turner

MINIMAL AREA GRAPH EMBEDDING INTO A DIRECTED GRID IS NP-HARD.. 234
H. M. Standley

ON I/O PLACEMENT IN VLSI CIRCUITS..................... 236
A. Aggarwal

EXACT ALGORITHMS FOR SPECIAL CASES OF THE BOARD PERMUTATION PROBLEM.................................. 246
J. Cohoon and S. Sahni

USING OVERLAP AND MINIMIZING CONTACT POINTS IN CHANNEL ROUTING.. 256
S. E. Hambrusch

ON SIMULATION AND INTERRELATIONS BETWEEN BOOLEAN AND VLSI CIRCUITS.. 258
A. Aggarwal

II-B: MULTI-USER INFORMATION THEORY

Organizer and Chairman: Edward van der Meulen
Katholieke Universiteit
Leuven, Belgium

A SIMPLIFIED TREATMENT OF WYNER'S WIRE-TAP CHANNEL.... 268
J. L. Massey

ERROR BOUNDS FOR ANTI-JAMMING CODES.................. 277
T. Ericson

AN ACHIEVABLE RATE REGION FOR THE MULTIPLE ACCESS CHANNEL WITH GENERALIZED FEEDBACK.................... 284
F. M. J. Willems, E. C. van der Meulen and J. P. M. Schalkwijk

THE BINARY AND GAUSSIAN RATE-DISTORTION FUNCTIONS FOR A DEGRADED SOURCE NETWORK........................ 293
T. Berger and C. Heegard

II-C: OPTIMAL CONTROL AND ROBOTICS

Chairman: Jose B. Cruz, Jr.
University of Illinois

THE CONTROL OF ROBOT MANIPULATORS USING AN OPTIMAL DECISION STRATEGY.................................... 303
M. W. Spong, S. Kheradpir and J. S. Thorp

ANALYSIS OF CONTROLLABILITY OF PLANAR TWO-LINK BIPED MODEL WITH MUSCLE ACTUATORS.................... 312
B. Bavarian

A DECENTRALIZED VARIABLE STRUCTURE CONTROL ALGORITHM FOR ROBOTIC MANIPULATORS............................ 315
R. G. Morgan and Ü. Özgüner

FEEDBACK OBSTACLE AVOIDANCE CONTROL.................. 325
B. H. Krogh

A SINGULAR PERTURBATION APPROACH TO THE DYNAMIC CONTROL OF ELASTIC ROBOTS............................ 335
A. Ficola, R. Marino and S. Nicosia

DECENTRALIZED SUCCESSIVE APPROXIMATION METHODS FOR OPTIMAL CONTROL.................................... 343
J. L. Calvet

II-D: SYSTEM-LEVEL DIAGNOSIS

Chairman: Gernot A. Metze
University of Illinois

A DIAGNOSIS ALGORITHM FOR THE BGM SYSTEM LEVEL FAULT MODEL.. 345
G. G. L. Meyer

ADAPTIVE FAULT IDENTIFICATION IN TWO NEW DIAGNOSTIC MODELS.................................... 353
S. E. Kreutzer and S. L. Hakimi

A PARTIAL ORDERING FOR SYSTEM-LEVEL FAULT MODELS...... 363
B. L. Havlicsek and G. G. L. Meyer

ON ADAPTIVE FAULT IDENTIFICATION IN OPTIMAL DIAGNOSABLE SYSTEMS.................................. 373
K. Nakajima and V. P. Krothapalli

THE DIAGNOSTIC CONFIDENCE OF T-FDOR SYSTEMS.......... 383
I. S. Hwang and C. R. Kime

III-A: COMPUTER AIDED DESIGN OF CONTROL SYSTEMS

Organizer and Chairman: S. P. Bingulac
University of Belgrade
Yugoslavia

RECENT MODIFICATIONS OF THE L-A-S (LINEAR ALGEBRA AND SYSTEMS) LANGUAGE AND ITS USE IN CAD OF CONTROL SYSTEMS........ 393
S. P. Bingulac

EXPERIENCE IN APPLYING A CONTROL DESIGN TECHNIQUE TO A POWER PLANT CONTROL PROBLEM........ 403
A. S. Brower, S. H. Javid and J. R. Winkelman

COMPUTER-AIDED DESIGN OF DIGITAL CONTROL USING PROJECTIVE CONTROLS........ 413
N. Gluhajić, J. Medanić and D. Petranović

COMPUTER-AIDED DESIGN OF CONTROL SYSTEMS: AN EXAMPLE OF WORKING SOFTWARE........ 423
S. P. Bingulac and P. J. West

SYSTOLIC ARRAYS FOR SOLVING LYAPUNOV AND RICCATI EQUATIONS........ 425
C. Bozzo, M. Fouques and E. Siffredi

III-B: COMPUTATIONAL GEOMETRY

Chairman: Franco P. Preparata
University of Illinois

OPTIMAL ALGORITHMS FOR COMPUTING DEPTHS AND LAYERS........ 427
B. Chazelle

A SIMPLE HIDDEN-LINE ALGORITHM FOR RECTILINEAR POLYGONS........ 437
J.-R. Sack

TRIANGULATING A POLYGON BY DIVIDE-AND-CONQUER........ 447
B. Chazelle and J. Incerpi

AN OPTIMAL ALGORITHM FOR COMPUTING THE MINIMUM VERTEX DISTANCE BETWEEN TWO CROSSING CONVEX POLYGONS........ 457
G. T. Toussaint

ON THE DUALITY OF INTERSECTIONS AND CLOSEST POINTS........ 459
C. Bajaj and M. Li

AN $O(|V|)$ PARALLEL ALGORITHM TO CONSTRUCT MINIMAL SPANNING TREES BASED UPON VORONOI DIAGRAMS........ 461
R. C. Chang and R. C. T. Lee

COMPUTING LARGEST EMPTY CIRCLES WITH LOCATION CONSTRAINTS........ 471
G. T. Toussaint

III-C: QUANTIZATION AND CODING

Chairman: Michael B. Pursley
University of Illinois

A FURTHER INVESTIGATION OF THE LLOYD-MAX ALGORITHM FOR QUANTIZER DESIGN.......... 481
F.-S. Lu and G. L. Wise

UNIFORM SPHERICAL COORDINATE QUANTIZERS.......... 491
P. F. Swaszek

A VECTOR LAPLACIAN QUANTIZER.......... 501
T. R. Fischer

ON THE PERMUTATION-DECODABILITY OF TRIPLE-ERROR-CORRECTING CODES.......... 511
S. G. S. Shiva, A. Benyamin-Seeyar and
V. K. Bhargava

PARITY DEFINITION BY OPERATIONS MODULO SOME 2^i-1 AND ITS POSSIBLE APPLICATION IN EVALUATING THE COMPLEXITY OF A LOG OPERATION OVER $GF(2^n)$.......... 520
B. Arazi

FINITE DECIPHERABILITY OF MULTIVALUED ENCODINGS.......... 528
R. M. Capocelli and U. Vaccaro

ERROR DETECTION, A GEOMETRIC APPOACH.......... 537
R. K. Ward

III-D: NONLINEAR SYSTEMS AND CONTROL

Chairman: Tzhy-Jong Tarn
Washington University

EVALUATION OF STEADY-STATE RESPONSES OF MILDLY NONLINEAR SYSTEMS.......... 546
I. W. Sandberg

GEOMETRIC STRUCTURE AND CONTROL DESIGN FOR A CLASS OF NONLINEAR SYSTEMS.......... 554
D. K. Lindner

OBSERVERS FOR NONLINEAR/TIME-VARYING SYSTEMS.......... 564
E. Emre

ON NULL CONTROLLABILITY OF NONLINEAR SYSTEMS ARISING IN AMCD-SPACECRAFT ATTITUDE CONTROL SYSTEM.......... 573
A. S. C. Sinha

PERIODIC EQUILIBRIA FOR THE MATRIX RICCATI EQUATION WITH PERIODIC COEFFICIENTS.......... 582
M. A. Shayman

IV-A: DETECTION AND ESTIMATION

Chairman: Kenneth S. Vastola
Rensselaer Polytechnic Institute

ROBUST MINIMAX LOCATION PARAMETER ESTIMATION UNDER UNIFORM LOSS WITH A COMPACT PARAMETER SPACE.......... 592
M. Zeytinoğlu and M. Mintz

ROBUST ESTIMATION AND DETECTION OF SIGNALS WITH ARBITRARY PARAMETERS.................................. 602
P. A. Kelly

FAULT TOLERANT PARALLEL IMPLEMENTATIONS OF THE KALMAN FILTER.. 610
G. G. L. Meyer and H. L. Weinert

OPTIMAL SEQUENTIAL SMOOTHING WITH UNCERTAIN OBSERVATIONS.. 613
H. Derin and M. Aşkar

SIGNAL DETECTION WHEN DEPENDENCE IS NEGLECTED......... 622
D. R. Halverson and G. L. Wise

DETECTION AND INFORMATION............................ 631
A. B. Martinez

IV-B: DATA STRUCTURES

Chairman: David E. Muller
University of Illinois

SELF-ORGANIZING HEURISTICS, IMPLICIT DATA STRUCTURES, AND UNSUCCESSFUL SEARCH 637
G. N. Frederickson

ON THE WORST-CASE EFFICIENT IMPLEMENTATION OF WEIGHTED DYNAMIC DICTIONARIES........................ 647
V. K. Vaishnavi

SURPRISINGLY EFFICIENT SEARCH ALGORITHMS FOR NONUNIFORMLY GENERATED FILES......................... 656
D. E. Willard

PREDICATE RETRIEVAL THEORY............................ 663
D. E. Willard

SOME PROPERTIES OF MULTI-ATTRIBUTE FILE SYSTEMS BASED UPON MULTIPLE KEY HASHING FUNCTIONS.................. 675
C. C. Chang and D. H. Su

NEW VARIANTS OF COALESCED HASHING.................... 683
W.-C. Chen and J. S. Vitter

IV-C: ADAPTIVE CONTROL AND STOCHASTIC SYSTEMS

Chairman: Petar V. Kokotovic
University of Illinois

AN EXPLICIT SELF-TUNING ALGORITHM BASED ON NONPARAMETRIC IDENTIFICATION........................... 693
R. Bars, L. Keviczky and K. S. P. Kumar

DESIGN OF A CONTINUOUS-TIME ADAPTIVE CONTROLLER FOR AN ANAEROBIC FERMENTATION PROCESS..................... 703
P. A. Ioannou and Ph. Opdenacker

THE STATISTICAL PROPERTIES OF THE RESIDUAL IN SYSTEM IDENTIFICATION................................ 712
J. M. Flentz, Y. F. Huang and W. J. Gadja, Jr.

CORRECTIVE METHODS FOR IDENTIFICATION IN THE CLOSED INFERENTIAL CONTROL LOOP.............................. 722
C. Wu and J. Gertler

ASYMPTOTICAL STABILITY OF RANDOM SYSTEMS............. 732
S. Chen

STOCHASTIC OPTIMAL CONTROL WITH FORCED SWITCHING ACTIONS... 734
S. A. Belbas and S. M. Lenhart

IV-D: CONTROLLER DESIGN TECHNIQUES

Chairman: Juraj Medanić
University of Illinois

FREQUENCY DOMAIN CONDITIONS FOR THE GENERAL SERVOMECHANISM PROBLEM................................ 744
D. Ohm, S. P. Bhattacharyya and J. W. Howze

ON THE TRACKING CONTROLLER DESIGN.................... 752
J. Kim and D. H. Chyung

THE COEFFICIENT WORDLENGTH PROBLEM IN DECENTRALIZED/HIERARCHICAL MULTIPROCESSOR CONTROL SYSTEMS....................................... 754
M. Burns and Ü. Özgüner

TIME DOMAIN DESIGN OF ROBUST CONTROLLERS FOR LINEAR REGULATORS...................................... 756
R. K. Yedavalli

MINIMUM NORM POLE ASSIGNMENT VIA SYLVESTER'S EQUATION.. 758
L. H. Keel, J. A. Fleming and S. P. Bhattacharyya

DESIGN AND IMPLEMENTATION OF PULSE MODULATED CONTROLLERS USING A MICROCOMPUTER.................... 760
C. P. Diduch and R. Balasubramanian

PERFORMANCE CRITERION SENSITIVITY OF SHIP STEERING MODELS-ANALYSIS AND DESIGN IMPLICATIONS...... 770
A. K. Tuǧcu, O. Coşkunoǧlu and R. E. Reid

MULTIVARIABLE CONTROL OF FLEXIBLE STRUCTURE - A PRELIMINARY STUDY.................................... 772
M. D. Lodaya and G. G. Leininger

A SIMPLIFIED ALGORITHM FOR TESTING STABILITY OF DISCRETE SYSTEMS..................................... 774
M. Ismail and H. K. Kim

IV-E: TESTING AND FAULT TOLERANCE

Chairman: Janak H. Patel
University of Illinois

SERIAL MULTIPLE FAULT DETECTION IN BANYAN NETWORKS.... 776
E. Opper and M. Malek

TEST DERIVATION FOR MOS SWITCH LOGIC NETWORKS......... 786
D. Radhakrishnan and G. K. Maki

COMMENTS ON A DATA COMPRESSION TECHNIQUE FOR VLSI TESTING... 796
K. K. Saluja and R. Dandapani

A SIMPLIFIED METHOD FOR TESTING MICROPROCESSORS....... 806
H. Singh and J. G. Bredeson

PLA IMPLEMENTATION OF K-OUT-OF-N CODE TSC CHECKER..... 812
B. Bose and D. J. Lin

ANALYSIS AND IMPLEMENTATION OF STATIC AND DYNAMIC FAULT TOLERANT RING SYSTEMS............................ 822
V. O. Roda, O. J. Davies and F. Lombardi

A SELECTIVELY REDUNDANT MULTIPROCESSOR SYSTEM FOR REAL TIME CONTROL...................................... 832
F. Ozguner, A. Ip and O. V. Dungern

V-A: PARALLEL AND DISTRIBUTED COMPUTATION

Chairman: Michael C. Loui
University of Illinois

MINIMIZING COMMUNICATION COST IN DISTRIBUTED QUERY PROCESSING IS NP-COMPLETE.............................. 842
K.-T. Huang

AN EVALUATION MODEL FOR INTERCONNECTION NETWORKS OF LARGE DISTRIBUTED PROCESSING SYSTEMS.................. 852
G. Cioffi and M. Lucertini

SORTING ON GATED-BUS ARRAY PROCESSORS................ 863
M.-Y. Chern and T. Murata

EXTENDED REGULAR EXPRESSIONS AND THEIR APPLICATION TO SPECIFICATION AND VERIFICATION OF DECENTRALIZED DAISY CHAIN ARBITERS.. 866
I. Suzuki, Y. Motohashi, K. Taniguchi, T. Kasami and T. Okamoto

COMPLEXITY OF OPTIMAL DEADLOCK RECOVERY.............. 876
J. Y-T. Leung

PREEMPTIVE SCHEDULING OF UNIFORM PROCESSORS WITH MEMORY.. 886
T.-H. Lai and S. Sahni

V-B: LARGE SCALE SYSTEMS AND MODEL REDUCTION

Chairman: Hassan K. Khalil
Michigan State University

DECENTRALIZED ESTIMATORS FOR LARGE GENERATION/LOAD DISTURBANCES IN INTERCONNECTED POWER SYSTEMS.......... 896
N. Čukalevski, M. Calović and J. Medanić

STRUCTURALLY STABLE FEEDBACK CONTROL OF SINGULAR SYSTEMS... 908
D. Cobb

SPECTRAL ASSIGNMENT USING REDUCED-ORDER MODELS........ 911
P. I. Carraway and R. R. Mielke

AN ITERATIVE ALGORITHM FOR OUTPUT FEEDBACK POLE PLACEMENT... 920
S. P. Bingulac and V. Mitrović

IMPACT OF EXPLICIT AND IMPLICIT CONTROL SHARING ON THE PERFORMANCE OF TWO-PERSON ONE-ACT LQG NASH GAMES.. 930
M. Tu and G. P. Papavassilopoulos

REDUCED-ORDER CONTROL FOR DISCRETE-TIME SYSTEMS WITH SLOW AND FAST MODES............................... 933
M. T. Tran and M. E. Sawan

ON A REACTIVE POWER FAULT MODELING IN LARGE SCALE ELECTRIC POWER SYSTEMS............................... 935
M. Ilic-Spong and M. Varghese

V-C: DIGITAL COMMUNICATIONS

Chairman: Dilip V. Sarwate
University of Illinois

EFFECT OF AMPLITUDE AND BANDLIMITING ON MULTI-H CPM DATA TRANSMISSION.................................. 945
M. A. Wickert and R. E. Ziemer

Q^2PSK - A NEW SPECTRALLY EFFICIENT MODULATION SCHEME.. 954
D. Saha

FIRST ORDER STATISTICAL CHARACTERISATION OF CYCLOSTATIONARY PROCESSES........................... 964
J. J. O'Reilly and W. J. Jun

TONE JAMMER CANCELLATION IN A DIRECT SEQUENCE CORRELATOR.. 971
G. R. Cooper and E. H. Cooper

A MAP APPROACH TO JOINT PHASE TRACKING-SYMBOL ESTIMATION FOR MULTI-H SIGNALS USING THE VITERBI ALGORITHM.................................... 972
J. M. Liebetreu and R. E. Ziemer

THE NOISE PENALTY IN APD RECEIVERS OPERATING WITH NON-OPTIMUM GAIN.................................... 981
L. J. Mayes

V-D: DIGITAL SIGNAL PROCESSING

Chairman: David C. Munson, Jr.
University of Illinois

STABILITY ANALYSIS OF FIXED-POINT DIGITAL FILTERS USING COMPUTER GENERATED LYAPUNOV FUNCTIONS.......... 983
K. T. Erickson and A. N. Michel

FIR DIGITAL FILTERS WITH INCREMENTAL COEFFICIENT VALUES.. 993
N. Benvenuto, L. E. Franks and F. S. Hill, Jr.

VLSI IMPLEMENTATION OF RNS MODULES AND THEIR PROPOSED APPLICATIONS............................ 1003
M. A. Bayoumi, G. A. Jullien and W. C. Miller

SOME STATISTICAL PROPERTIES OF GENERALIZED MEDIAN FILTERS... 1014
Y. H. Lee and S. A. Kassam

ROOT PROPERTIES AND CONVERGENCE RATES OF MEDIAN FILTERS... 1024
J. P. Fitch, E. J. Coyle and N. C. Gallagher, Jr.

MEDIAN FILTERS: ANALYSIS FOR SIGNALS WITH ADDITIVE IMPULSE NOISE...................................... 1034
R. J. Crinon and G. R. Arce

SOLVING ILL-CONDITIONED SYSTEMS USING SINGULAR VALUE DECOMPOSITION WITH APPLICATION TO SIGNAL EXTRAPOLATION.. 1044
B. J. Sullivan and B. Liu

APPROXIMATING THE EIGENVECTORS OF A SYMMETRIC TOEPLITZ MATRIX.................................... 1046
D. R. Fuhrmann and B. Liu

A NOTE ON STATE TRAJECTORIES OF CAUSAL INPUT-OUTPUT SYSTEMS

WILLIAM L. ROOT
Aerospace Engineering Department
The University of Michigan
Ann Arbor, MI 48109

ABSTRACT

State-space and state are defined for a class of causal, time-invariant, input-output systems described abstractly in terms of fitted families of normed linear spaces. It is shown that the state is continuous with respect to past history of the input, and that the state trajectory is continuous.

INTRODUCTION

This is an essentially self-contained note on state and state trajectories for causal, time-invariant, input-output systems, described within a certain abstract framework which I believe to be theoretically useful. This framework was introduced in [1] for the study of some properties of time-varying systems – in particular the characteristics of what are called system trajectories; it is used in [2] and [3] for the study of system trajectories of stochastic systems.

The reason for looking once again at the well worked-over concept of state is simply that, in the abstract setting alluded to, a definition and an elementary theory have not previously been developed. It seems reasonable to consider first the time-invariant case, and that is all that is considered here. The results are definitions of state-space and state, which require some preliminary verifications, a theorem on continuity of state with respect to past history, and a theorem on continuity of state trajectories. The proofs are elementary in the technical sense and are also easy; they are given in some detail because they involve unconventional rules of manipulation.

PRELIMINARIES

For the purposes of this paper, a *system* (Y,F,U) is a mapping $F:U \rightarrow Y$. Each $u \in U$ represents a possible input or stimulus, and the element $y=F(u)$ represents the corresponding output. Both U and Y will be function spaces defined on the real line; i.e. each $u \in U$, $y \in Y$ will be a space whose elements are functions, or equivalence classes of functions, of a real variable (time). These functions may be vector-valued. These statements hold also when any of various suffixes are attached to U and Y.

We say U(Y) is *shift-invariant* if U(Y) is a function space on R^1, and $f \varepsilon U(Y)$ implies any translate of $f \varepsilon U(Y)$. The symbol L_τ is used to denote translation to the left by τ; the notation $R_\tau = L_{-\tau}$ is also used. If a space is shift-invariant, L_τ is a bijection.

If U is a metric space and Y is a Banach space, we define $\mathcal{F}(U,Y)$ to be the Banach space of all bounded continuous mappings $F: U \rightarrow Y$ with norm given by

$$||F|| \overset{\Delta}{=} \sup_{u \varepsilon U} ||F(u)||_Y.$$

An abstract, rather general, model for causal systems is constructed below and used in all that follows. This model requires the notion of fitted-families of normed linear spaces for the definitions of both input and output spaces. Fitted-families of normed linear spaces are introduced and defined in [1], but for convenience the definition is repeated here in the Appendix. Actually, the last part of the definition given in the Appendix, that of extended fitted families, is new; it was not needed for previous work, but it is here. There is a very large class of examples of fitted families; see [1].

We suppose $(\mathcal{L},N)$ and $(\mathcal{L}',N')$ are given; these determine extended fitted families $\{A_{s,t}\}$ and $\{B_{s,t}\}$ respectively. Assume the $\{B_{s,t}\}$, $t < \infty$, are Banach spaces (i.e. they are complete). As a notational convenience let $||\cdot||_{s,t}$ and $|||\cdot|||_{s,t}$ denote the norms in $A_{s,t}$ and $B_{s,t}$, respectively.

Definition 1. Let U be a shift-invariant subset of A, the bounding space for $\{A_{s,t}\}$, and $Y = B^e$. Let $F: U \rightarrow Y$ satisfy the condition: for all $u \varepsilon U$ and all $t < \infty$, $||u'-u''||_t = 0$ implies $|||F(u')-F(u'')|||_t = 0$. Then (Y,F,U) is a *causal system (CS) with respect to* $\{A_{s,t}\}$ *and* $\{B_{s,t}\}$, and F is a *causal mapping.* ||

We denote by $U_{s,t}$ the metric subspace of $A_{s,t}$ determined by U. By the definition, a causal F induces a mapping $\tilde{F}_t: U_t \rightarrow B_t$; F also induces a mapping $\tilde{F}_{s,t}: U_t \rightarrow B_{s,t}$ for any $s \leq t$. The CS (Y,F,U) is *continuous (bounded)* if all the $\tilde{F}_{s,t}$ are continuous (bounded). If all the $\tilde{F}_t$ are continuous or bounded, so are all the $\tilde{F}_{s,t}$, $s \leq t < \infty$. Let $0 \leq b \leq \infty$ be fixed, then in any CS there is a one-parameter family of mappings $\{\tilde{F}_{t-b,t}: U_t \rightarrow B_{t-b,t}\}$ determined by F. Define $F_{(b)t}: U_o \rightarrow B_{-b,o}$ by $F_{(b)t}(u_o) \overset{\Delta}{=} L_t \tilde{F}_{t-b,t}(R_t u_o)$, $u_o \varepsilon U_o$. The correspondence between $F_{(b)t}$ and $\tilde{F}_{t-b,t}$ is 1:1. If the CS (Y,F,U) is continuous and bounded, then $F_{(b)t} \varepsilon \mathcal{F}(U_o,B_{-b,o})$ and the correspondence is an isometric isomorphism. The CS (Y,F,U) is *time-invariant* iff $F = L_s F R_s$ for all s. Obviously, in this case $F_{(b)o} = F_{(b)t}$ for all t.

So-called *system trajectories*, the mappings $t \to F_{(b)t}$ for any fixed b, are investigated in [1], and for stochastic systems in [2] and [3]. Since system trajectories so defined are trivial in the case of time-invariant systems, they do not concern us further here. In fact they have been mentioned only so that it may be emphasized that system trajectories and state trajectories are *not* the same thing.

In what follows, to save notation, we assume the output spaces are always B_t ($B_{-\infty,t} = B_t$); then we can write F_t for $F_{(b)t}$. This involves no real loss of generality.

STATE AND STATE TRAJECTORY

Input Spaces: Conditions

In order to define state we need to introduce an operation on input functions to be called splicing, and then require that the fitted family $\{\mathcal{L},N\}$ be such that splicing is always possible.

Let $f \in \mathcal{L}_{oo}$, then for $-\infty \le r < s \le \infty$, $f_{r,s}$ is the function defined on $(r,s]$ that agrees with f. Suppose for $g,h \in \mathcal{L}_{oo}$ there is a function $f \in \mathcal{L}_{oo}$ such that

$$\begin{aligned} f_{r,t}(\tau) &= g_{r,s}(\tau), \quad r < \tau \le s \\ &= h_{s,t}(\tau), \quad s < \tau \le t, \end{aligned}$$

then we write $f_{r,t} = g_{r,s} \leftrightarrow h_{s,t}$. Further, let $u_{r,t} \in A_{r,t}$, $v_{r,s} \in A_{r,s}$, $w_{s,t} \in A_{s,t}$ be determined by f, g, h respectively; then we write $u_{r,t} = v_{r,s} \leftrightarrow w_{s,t}$. This operation is the splicing operation; $u_{r,t}$ so defined is the *splice* of $v_{r,s}$ and $w_{s,t}$. In general, splicing cannot always be done (e.g., if $\mathcal{L}$ is a space of continuous functions), but for a large class of fitted families $(\mathcal{L},N)$ it can. We assume for what follows that the shift-invariant input set U is contained in A, where A is the bounding space for a fitted family $(\mathcal{L},N)$ that allows splicing for arbitrary v,s and arbitrary $-\infty \le r < s < t \le \infty$.

Note that if $h,h' \in v_{r,s}$ and $g,g' \in w_{s,t}$, $r < s < t$, then

$$\begin{aligned} \| h_{r,s} \leftrightarrow g_{s,t}) - (h'_{r,s} \leftrightarrow g'_{s,t})\|_{r,t} &\le \|\ldots\|_{r,s} + \|\ldots\|_{s,t} \\ &= \|h-h'\|_{r,s} + \|g-g'\|_{s,t} = 0. \end{aligned}$$

Thus, when splicing is permissible $v_{r,s} \leftrightarrow w_{s,t}$ is always well-defined. However, while $v_{r,s} \leftrightarrow w_{s,t} = v'_{r,s} \leftrightarrow w'_{s,t}$ always implies $w_{s,t} = w'_{s,t}$, it does *not* imply $v_{r,s} = v'_{r,s}$ (unless $\alpha = \infty$, see Appendix). Thus a decomposition $u_{r,t} = u_{r,s} \leftrightarrow u_{s,t}$ is not in general unique in the first term. Clearly, $u_{r,s} \leftrightarrow v_{s,t} \leftrightarrow w_{t,u}$, $r < s < t < u$, is well-defined, as is a splice with any finite number of sections. Also

$L_a(u_{r,s} +\!\!\rightarrow v_{s,t}) = L_a u_{r,s} +\!\!\rightarrow L_a v_{s,t}$.

We also require for what is to follow that the time-invariant map $F: U \rightarrow B^e$ is such that $F_o: U_o \rightarrow B_o$ is uniformly continuous and bounded. If $U = A$, this rules out linear mappings F, of course. If one wants to have $U = A$ and include the linear maps, then F cannot be required to induce an F_o that belongs to $\mathcal{F}(U_o, B_o)$; presumably this situation can be handled using a different topology on the space of mappings, but it is not considered here. At the moment, I am more interested in the case U is bounded – then F can be a restriction of a linear mapping. In this case there is an obvious technical difficulty with splicing: $u_{r,s} \in U_{r,s}$ and $v_{s,t} \in U_{s,t}$ does not imply that $u_{r,s} +\!\!\rightarrow v_{s,t} \in U_{r,t}$. To circumvent this difficulty we suppose that although the actual inputs are constrained to lie in U, F is defined and is bounded and uniformly continuous on a set U* that is bigger than U and contains U. Precisely, let U* be a shift-invariant subset of A such that:

i) $||u||_t \leq c$, $t \in R^1$, $u \in U^*$, for some $0 < c < \infty$,

ii) $u \in U^*$ implies $au \in U^*$, where a is any real number, $0 \leq a \leq 1$.

We say U* is *full* if (ii) is satisfied. Then let

$$U = \{u \in U^*: ||u||_t \leq c/2, \quad t \in R^1\};$$

we write $U = 1/2\, U^*$. Clearly, U is bounded in A and is shift-invariant, and $u_{r,s} \in U_{r,s}$, $v_{s,t} \in U_{s,t}$ implies $u_{r,s} +\!\!\rightarrow u_{s,t} \in U^*_{r,t}$.

A Definition of State

Definition 2. We say a system (Y,F,U) satisfies *condition (S)* if the following requirements are met:

(1) (Y,F,U) is a CS in the sense of Definition 1, and is time-invariant.

(2) A, the bounding space for $(\mathcal{L}, N)$, permits splicing.

(3) U* is a full, bounded, shift-invariant subset of A and $U = 1/2\, U^*$.

(4) There is a map $F': U^* \rightarrow B^e$ such that the induced map $F'_o: U^*_o \rightarrow B_o$ is bounded and uniformly continuous. F is the restriction of F' to U.||

The map $F_o: U_o \rightarrow B_o$ induced by F may also be regarded as a restriction of F'_o. Henceforth, the primes are dropped and the notations F, F_o, $\tilde{F}_t$ can refer to either the restricted or nonrestricted mappings.

Lemma 1. Under the conditions just stated, F is actually a mapping from U* into B. It is bounded and uniformly continuous.

Proof. Using some properties of fitted families (see Appendix) and time-invariance, one has for $u \varepsilon U^*$:

$$\begin{aligned} |||F(u)||| &= \sup_s |||F(u)|||_s = \sup_s |||R_s F L_s(u)|||_s \\ &= \sup_s |||F_o(L_s u|||_o \\ &\leq \sup_{U_o} |||F_o(u_o)|||_o = ||F_o|| . \end{aligned}$$

Similarly, for $u,v \varepsilon U^*$:

$$\begin{aligned} |||F(u) - F(v)||| &= \sup_s |||F(u) - F(v)|||_s \\ &= \sup_s |||R_s F\ L_s(u) - R_s F\ L_s(v)|||_s \\ &= \sup_s |||F_o(L_s u)_o - F_o(L_s v)_o|||_o . \qquad (1) \end{aligned}$$

Also, $$\begin{aligned} ||(L_s u)_o - (L_s v)_o||_o &= ||L_s(u-v)||_o \\ &= ||u-v||_s \leq ||u-v|| . \end{aligned}$$

Now, given $\varepsilon > 0$ there is $\delta > 0$ such that $||w_o - w'_o||_o \leq \delta$ implies $|||F_o(w_o) - F_o(w'_o)|||_o \leq \varepsilon$. Hence, $||u-v|| \leq \delta$ implies $|||F(u)-F(v)||| \leq \varepsilon$ by Eq. (1).||

Lemma 2. Under the same conditions as for Lemma 1, for arbitary fixed $p \varepsilon R^1$ and arbitrary fixed $u_p \varepsilon U_p$, there exists a bounded uniformly continuous mapping $\tilde{G}^u_p: U_{p,\infty} \to B_{p,\infty}$ such that for all $v_{p,\infty} \varepsilon U_{p,\infty}$, $\tilde{G}^u_p(v_{p,\infty})$ is the same element of $B_{p,\infty}$ as that determined by $F(u_p +\!\!\to v_{p,\infty})$.

Proof. Since by Lemma 1, F maps U* into B, and $|||F(u_p +\!\!\to v_{p,\infty})|||_{p,\infty} \leq |||F(u_p +\!\!\to v_{p,\infty})|||_\infty$ for all $u_p \varepsilon U_p$, $v_{p,\infty} \varepsilon U_{p,\infty}$, it is clear that $F(u_p +\!\!\to v_{p,\infty})$ does indeed determine an element in $B_{p,\infty}$. We define $\tilde{G}^u_p(v_{p,\infty})$ to be that element, so we may write (see Appendix)

$$|||\tilde{G}^u_p(v_{p,\infty}) - F(u_p +\!\!\to v_{p,\infty})|||_{p,\infty} = 0.$$

Then, since

$$|||F(u_p +\!\!\to v_{p,\infty})|||_{p,\infty} \leq |||F(u_p +\!\!\to v_{p,\infty})|||_\infty \leq ||F|| ,$$

$$|||\tilde{G}^u_p(v_{p,\infty})|||_{p,\infty} \leq ||F|| .$$

Similarly, for $u_p \varepsilon U_p$ and $v_{p,\infty}$, $v'_{p,\infty} \varepsilon U_{p,\infty}$,

$$\begin{aligned} &|||\tilde{G}^u_p(v_{p,\infty}) - \tilde{G}^u_p(v'_{p,\infty})|||_{p,\infty} \\ &\qquad \leq |||F(u_p +\!\!\to v_{p,\infty}) - F(u_p +\!\!\to v'_{p,\infty})||| \end{aligned}$$

Now, $$\begin{aligned} ||(u_p +\!\!\to v_{p,\infty}) - (u_p +\!\!\to v'_{p,\infty})|| &\leq ||\ldots||_p + ||\ldots||_{p,\infty} \\ &= ||v_{p,\infty} - v'_{p,\infty}||_{p,\infty} . \end{aligned}$$

Hence, given $\varepsilon > 0$ there is $\delta > 0$ such that $\| v_{p,\infty} - v'_{p,\infty} \|_{p,\infty} \le \delta$ implies $|||\tilde{G}^u_p(v_{p,\infty}) - \tilde{G}^u_p(v'_{p,\infty})|||_{p,\infty} \le \varepsilon$.‖

Definition 3. The *complete state* for a system (Y,F,U), satisfying condition (S), for any time t and any $u_t \in U_t$, is defined to be the mapping $G^u_t : U_{0,\infty} \to B_{0,\infty}$ given by $G^u_t \overset{\Delta}{=} L_t \tilde{G}^u_t R_t$. The *state* is G^u_t restricted to those $v_{0,\infty} \in U_{0,\infty}$ for which $(u_t \mathbin{+\!\!\!\to} R_t v_{0,\infty}) \in U$.‖

The reason for considering the complete state G^u_t is obviously that each G^u_t has the same domain, whereas its restriction has domain depending on u_t. By Lemma 2, $G^u_t \in \mathcal{F}(U_{0,\infty},B_{0,\infty})$, which we define to be the state space.

Remark 1. The state at time s given u_s, and an arbitrary $u_{s,t}$, $s < t$, such that $u_t \overset{\Delta}{=} u_s \mathbin{+\!\!\!\to} u_{s,t} \in U_t$, determine the complete state at time t given u_t. In fact, for any $v_{t,\infty} \in U_{t,\infty}$, $|||\tilde{G}^u_s(u_{s,t} \mathbin{+\!\!\!\to} v_{t,\infty}) - F(u_s \mathbin{+\!\!\!\to} u_{s,t} \mathbin{+\!\!\!\to} v_{t,\infty})|||_{s,\infty} = 0$ and $|||F(u_s \mathbin{+\!\!\!\to} u_{s,t} \mathbin{+\!\!\!\to} v_{t,\infty}) - \tilde{G}^u_t(v_{t,\infty})|||_{t,\infty} = 0$. Also, with y_s known, they determine the output at time t, i.e. $y_t \in B_t$, produced by the input $u_t \in U_t$. These comments justify the definition of state. Furthermore, a simple verification shows that the complete state at time t depends only on the input up to time t and not on t itself. That is, if $v = L_a u$, $a \in R^1$, $G^u_t = G^v_{t-a}$.‖

Because of the generality and nonparametric form of the definition, it seems fair to refer to it as giving the state in canonical form for systems as specified. It may well happen that the state space $\mathcal{F}(U_{0,\infty},B_{0,\infty})$ is larger than necessary. From Remark 1 it follows that if the closed linear span of all G^u_t with t fixed is a proper subspace $\mathfrak{M}$ of $\mathcal{F}(U_{0,\infty},B_{0,\infty})$, then the same is true for all G^u_s, $s \ne t$, and $\mathfrak{M}$ is a reduced state space.

Continuity of state and of state trajectory

Theorem 1. The complete state at time t for a system that satisfies condition (S) is a uniformly continuous function of the input prior to t. That is, the mapping $u_t \to G^u_t$ is uniformly continuous with respect to the metrics of A_t and $\mathcal{F}(U_{0,\infty},B_{0,\infty})$.

Proof. By definition, for $u_t, u'_t \in U_t$

$$\begin{aligned}
\| G^u_t - G^{u'}_t \| &= \| L_t \tilde{G}^u_t R_t - L_t \tilde{G}^{u'}_t R_t \| \\
&= \sup_{U_{0,\infty}} |||L_t[\tilde{G}^u_t R_t(w_{0,\infty}) - \tilde{G}^{u'}_t R_t(w_{0,\infty})]|||_{0,\infty} \\
&= \sup_{U_{0,\infty}} |||\tilde{G}^u_t(R_t w_{0,\infty}) - \tilde{G}^{u'}_t(R_t w_{0,\infty})|||_{t,\infty} \\
&= \sup_{U_{t,\infty}} |||\tilde{G}^u_t(v_{t,\infty}) - \tilde{G}^{u'}_t(v_{t,\infty})|||_{t,\infty}
\end{aligned}$$

$$\| G_t^u - G_t^{u'} \| = \sup_{U_{t,\infty}} |||F(u_t \leftrightarrow v_{t,\infty}) - F(u'_t \leftrightarrow v_{t,\infty})|||_{t,\infty} .$$

$$\leq \sup_{U_{t,\infty}} |||F(u_t \leftrightarrow v_{t,\infty}) - F(u'_t \leftrightarrow v_{t,\infty})||| . \qquad (2)$$

Now, $\|(u_t \leftrightarrow v_{t,\infty}) - (u'_t \leftrightarrow v_{t,\infty})\| \leq \|\ldots\|_t + \|\ldots\|_{t,\infty}$

$$= \|u_t - u'_t\|_t .$$

Hence by Lemma 1, given $\varepsilon > 0$ there is $\delta > 0$ such that $\|u_t - u'_t\|_t \leq \delta$ implies the right side of the inequality (2) is $\leq \varepsilon$.||

Definition 4. The *state trajectory given* $u \in U$ of a system with complete state G_t^u is the mapping $t \to G_t^u$.

Theorem 2. Suppose a system (Y,F,U) satisfies Hypothesis S, where $U \subset A$ and the left extended fitted family $\{A_{st}\}$ is shift continuous. Then the state trajectory given u, for any $u \in U$, is continuous.

Proof: It is necessary to show that

$$\| G_t^u - G_{t+h}^u \| = \sup_{v_{0,\infty} \in U_{0,\infty}} |||G_t^u(v_{0,\infty}) - G_{t+h}^u(v_{0,\infty})|||_{0,\infty}$$

$$\to 0 \quad \text{as } h \to 0.$$

One has, using $L_{t+h} = L_t L_h$,

$$|||G_t^u(v_{0,\infty}) - G_{t+h}^u(v_{0,\infty})\|_{0,\infty}$$

$$= ||| \tilde{G}_t^u(R_t v_{0,\infty}) - L_h \tilde{G}_{t+h}(R_{t+h}, v_{0,\infty})|||_{t,\infty}$$

$$= |||F(u_t \leftrightarrow R_t v_{0,\infty}) - L_h F(u_{t+h} \leftrightarrow R_{t+h} v_{0,\infty}|||_{t,\infty} . \qquad (3)$$

Since $L_h F(u_{t+h} \leftrightarrow R_{t+h} v_{0,\infty})$

$$= L_h F R_h(L_h u_{t+h} \leftrightarrow R_t v_{0,\infty})$$

$$= F(L_h u_{t+h} \leftrightarrow R_t v_{0,\infty})$$

by time-invariance, the right side of Eq. (3) can be rewritten as

$$|||F(u_t \leftrightarrow R_t v_{0,\infty}) - F(L_h u_{t+h} \leftrightarrow R_t v_{0,\infty})|||_{t,\infty}$$

$$= \sup_{t < s} \| \ldots \|_{t,s} \leq \sup_{t < s} \| \ldots \|_s . \qquad (4)$$

Put $w \triangleq u_t \leftrightarrow R_t v_{0,\infty}$, $z \triangleq L_h u_{t+h} \leftrightarrow R_t v_{0,\infty}$. Then the right side of the inequality (4) can be rewritten in the form

$$\sup_{t < s} |||\tilde{F}_s(w_s) - \tilde{F}_s(z_s)|||_s$$

$$= \sup_{t < s} |||F_0(L_s w)_0 - F_0(L_s z)_0|||_0 . \qquad (5)$$

By a now familiar argument, for $t < s$,

$$\| L_s w - L_s z \|_o = \| w - z \|_s$$

$$\le \| u_t - L_h u_{t+h} \|_t = \| f - L_h f \|_t \qquad (6)$$

for any $f \in \mathcal{L}_{oo}$ such that $f \in u$. By shift-continuity the right side of Eq. (6) tends to zero as $h \to 0$. Thus, given $\delta > 0$, there is $h(\delta)$, depending on u, such that $\| L_s w - L_s z \|_o \le \delta$ whenever $|h| \le h(\delta)$, for all $t < s$ and for all $v_{o,\infty} \in U_{o,\infty}$. By Eqs. (3) and (5) and the inequality (4),

$$\sup_{v_{o,\infty} \in U_{o,\infty}} ||| G^u_t(v_{o,\infty}) - G^u_{t+h}(v_{o,\infty}) |||_{o,\infty}$$

$$\le \sup_{v_{o,\infty} \in U_{o,\infty}} \sup_{t < s} ||| F_o(L_s w)_o - F_o(L_s z)_o |||_o. \qquad (7)$$

Then, since F_o is uniformly continuous, given $\varepsilon > 0$ there is $\delta > 0$ such that the right side of the inequality (7) is less than ε whenever $|h| \le h(\delta)$.||

Remark 2. A sufficient condition that the left-extended fitted family $\{A_{s,t}\}$ be shift-continuous is that the nonextended family be shift-continuous and tapered (see [1], Proposition 3).||

Remark 3. If U = A, clearly all the above results hold if (1) and (2) of condition (S) are satisfied and (4) is replaced by: the map $F: U \to B^e$ is such that the induced $F_o: U_o \to B_o$ is bounded and uniformly continuous.||

Remark 4. The condition that F_o is bounded is shown by Lemma 1 to be a stability condition for time-invariant systems, according to a reasonable notion of stability. Thus, all the above considerations are for stable systems.||

APPENDIX

Definition A1. (see [1]). Let $\mathcal{L} = \mathcal{L}(R,B)$ be a translation-invariant linear space of functions from R into a Banach space B. Let $N = \{\|\cdot\|_{s,t}, -\infty < s \le t < \infty\}$ be a family of seminorms on $\mathcal{L}$ satisfying:

(1) If $g_1, g_2 \in \mathcal{L}$ and $g_1(\tau) = g_2(\tau)$, $s \le \tau \le t$, then $\|g_1 - g_2\|_{s,t} = 0$.

(2) $\| L_\tau g \|_{s-\tau,t-\tau} = \| g \|_{s,t}$, $g \in \mathcal{L}$, $-\infty < \tau < \infty$

(3) If $r \le s \le t$, $\| g \|_{s,t} \le \| g \|_{r,t}$, $g \in \mathcal{L}$

(4) If $r \le s \le t$, $\| g \|_{r,t} \le \| g \|_{r,s} + \| g \|_{s,t}$, $g \in \mathcal{L}$

(5) For some $\alpha > 0$ (α may be $+\infty$) and $K \ge 0$, and for all $g \in \mathcal{L}$

$\| g \|_{r,s} \le K \| g \|_{r,t}$, $r \le s \le t$, whenever $t - r \le \alpha$.

The pair $(\mathcal{L},N)$ is called a *fitted family of seminorms on* $\mathcal{L}$. The normed linear space formed from equivalence classes of functions in $\mathcal{L}$ with norm $\|\cdot\|_{s,t}$ is denoted $A_{s,t}$ (or $B_{s,t}$). The set $\{A_{s,t}\}$ is the *fitted family of normed linear spaces* determined by $(\mathcal{L},N)$.||

Definition A2. With $(\mathcal{L},N)$ given, let

$$\mathcal{L}_o = \{g \in \mathcal{L}: \lim_{s \to -\infty} \|g\|_{s,t} \text{ exists, } t \in R\}.$$

Then $\mathcal{L}_o$ is linear and translation-invariant. For $g \in \mathcal{L}_o$, let $\|g\|_t \triangleq \lim_{s \to -\infty} \|g\|_{s,t}$; $\|\cdot\|_t$ is a seminorm on $\mathcal{L}_o$. Let $\bar{N} = \{\|\cdot\|_{s,t}, -\infty \leq s \leq t < \infty$, but $-\infty < t\}$, where $\|\cdot\|_{-\infty,t} = \|\cdot\|_t$. The properties (1),...,(5) above hold for $(\mathcal{L}_o,\bar{N})$, called the *left- extended fitted family of seminorms on* $\mathcal{L}_o$.

Normed linear spaces $A_t = A_{-\infty,t}$ are now defined, using $(\mathcal{L}_o,\bar{N})$, analogously to the $A_{s,t}$, and we call $\{A_{s,t}\}$, $-\infty \leq s \leq t < \infty$ but $-\infty < t$, the *left-extended fitted family on* $\mathcal{L}_o$. $\|$

Definition A3. Let $\mathcal{L}_{oo} \triangleq \{g \in \mathcal{L}_o: \sup_{t \in R} \|g\|_t < \infty\}$. $\mathcal{L}_{oo}$ is also linear and translation invariant. For $g \in \mathcal{L}_{oo}$ define

$$\|g\|_{s,\infty} \triangleq \sup_{s < t < \infty} \|g\|_{s,t}$$

and $\|g\| \triangleq \sup_{t < \infty} \|g\|_t$. These are seminorms on $\mathcal{L}_{oo}$. Let $\bar{\bar{N}} = \{\|\cdot\|_{s,t}, -\infty \leq s \leq t \leq \infty$, but $-\infty < t, s < \infty\}$ and put $\|\cdot\| = \|\cdot\|_{-\infty,\infty}$. The properties (1),...,(5) above hold for $(\mathcal{L}_{oo},\bar{\bar{N}})$, called the *extended fitted family of seminorms on* $\mathcal{L}_{oo}$.

Normed linear spaces $A_{s,\infty}$ are now defined, using $(\mathcal{L}_{oo},\bar{\bar{N}})$, analogously to the $A_{s,t}$, and we call $\{A_{s,t}\}$, $-\infty \leq s \leq t \leq \infty$, but $-\infty < t$, $s < \infty$, the *extended fitted family on* $\mathcal{L}_{oo}$. $\|$

We write $A = A_{-\infty,\infty}$, and call it the *bounding space* for $\{A_{s,t}\}$ (see [1]). The following inequalities follow immediately for $g \in \mathcal{L}_{oo}$:

$$\|g\|_{s,t} \leq \|g\|_t \leq \|g\|$$

and, $\|g\|_{s,t} \leq \|g\|_{s,\infty} \leq \|g\|$.

Note that $\|g\|_{s,\infty} \neq \lim_{t \to \infty} \|g\|_{s,t}$ in general, even when the limit exists.

The verifications of properties (1),...,(5) for the left-extended and extended fitted families are routine. In the paper [1] where the fitted family concept is introduced, left-extended families are called simply extended families, and what is called here the extended family is not defined.

Definition A4. Let A^e, the *expanded space* for $\{A_t\}$, be the linear space of equivalence classes of functions in $\mathcal{L}_o$, where $f,g \in \mathcal{L}_o$ are equivalent iff $\|f-g\|_t = 0$ for all t. A^e is not given a topology. $\|$

Note that $A^e \supset A$ and $\|\cdot\|_{s,t}$ is a seminorm on A^e. Consider the left-extended family $\{A_{s,t}\}$. Let $u_{s,t}$ be an element of the normed linear space $A_{s,t}$, and let $f \in \mathcal{L}_o$ be such that f belongs to the equivalence class $u_{s,t}$,

$f \varepsilon u_{s,t}$. Then f determines an equivalence class $u \varepsilon A^e$, and any $g \varepsilon u$ will also belong to $u_{s,t}$, since $\| f-g\|_{s,t} \leq \|f-g\|_t$. Thus $u \subset u_{s,t}$, and the expressions in the following equation, though inelegant, have a well-defined meaning

$$\| u_{s,t} - f\|_{s,t} = \| u_{s,t} - u\|_{s,t} = 0$$

Similarly, when $u_{s,}$ is defined

$$\| u_{s,t} - u_{s,\infty}\|_{s,t} = 0.$$

REFERENCES

[1] W. L. Root, "Considerations regarding input and output spaces for time-varying dynamical systems," *Appl. Math. Optim.*, 4: 365-384 (1978).

[2] W. L. Root, "Generic models for stochastic systems," in *Analysis and Optimisation of Stochastic Systems*, Jacobs, Davis, Demster, Harris and Parks, eds. Academic Press, London. 165-180 (1980).

[3] W. L. Root, "Randomized system trajectories," *Appl. Math. Optim.*, 8: 293-307 (1982).

ACKNOWLEDGEMENT

This research was supported by the National Science Foundation under Grant ECS-8005960.

A POLYNOMIAL FRACTIONAL APPROACH TO LINEAR TIME-VARYING SYSTEMS

E. EMRE
Department of Electrical Engineering/Computer Science
Texas Tech University
Lubbock, TX 79409

1. INTRODUCTION

Linear time-varying systems has been an active area of research for several researchers thus far with several approaches (see Kalman [1], Kalman and Bucy [2], Silverman and Meadows [3], Morse and Silverman [4], Silverman [5], D'angelo [6], Kalman, Falb and Arbib [7, Chpt. 10], Silverman [8], Kalmen [9], Kamen and Hafez [10], Emre [11], Kamen and Khargonekar [13], Kamen, Khargonekar and Poola [14], Ylinen [12], and the references given there).

Several algebraic similarities between the theory of linear time-invariant systems and linear time-varying systems have been reported, and some results in this direction have been established in several of the references above.

However, a general algebraic theory of control based on (polynomial) fractional representations of such systems which is applicable to both discrete-time as well as continuous-time case simultaneously has not appeared in the literature thus far.

In this paper, we establish an algebraic theory of control for linear time-varying systems which closely follows a corresponding theory for linear systems over rings given in Emre [15-17] and Emre and Khargonekar [18]. Similar to the theory in [15-18], this theory is also based on the existence of (left or right) division by certain polynomial operators, and we show that this enables us to study a linear time-varying system in terms of a corresponding formal time-invariant system which behaves very much like a linear system over a ring. We show that, in terms of this system, the results established in [15-18] translate to be applicable to linear time-varying systems to a considerable extent. In our framework, most of our results are applicable to discrete-time or continuous-time systems (over a ring of infinitely differentiable functions) exactly the same way, although one can obtain these results more easily for discrete-time systems. The main difficulty in such an algebraic theory for continuous-time systems is the following. In case of time-invariant systems, transfer matrices are well defined as formal power series in an indeterminate z^{-1}. For discrete-time time-varying systems, it is again easy to see that formal power series in p^{-1} (the leftshift operator) can be easily defined. However, for continuous time systems, since the derivative operator p does not have a unique left inverse with nice properties, there is thus far no known way of suitably defining formal power series and p^{-1}, with all the desirable properties that such series have in the linear time-invariant or discrete-time time-varying case. With such a well defined formal power series, most of the existing theory (see [15-18] and the references there) could be carried out much as in the time-invariant case. However, based on the observation that the main use of formal power series in these cases is to be able to define a certain projection, we are able to resolve this difficulty to a considerable extent. We establish a theory for continuous-

---

This research was supported in part by the U.S. Air Force Office of Scientific Research Grant AFOSR-82-0282 and in part by the National Science Foundation under NSF Research Grant ECS-8204512.

time systems which is based on the existence of a division algorithm for time-varying differential operators, bypassing the use of power series, which is equally applicable to discrete-time time-varying systems. One can translate these results to time-domain easily putting the discrete-time or continuous-time interpretations of these unified results. Thus, to a large extent, our theory also unifies the study of continuous-time and discrete-time linear time-varying systems with that of linear systems over rings as far as main control problems are concerned. Our definition of polynomial fractional descriptions of linear time-varying systems is the same as in the case of time-invariant systems, (e.g., of the usual form $Q(t,)y(t) = RE(t,p)u(t)$ where $Q(t,p)$ and $R(t,p)$ are time-varying differential or difference operators).

The results established here are summarized as follows.

In Section 2, we first consider a monic time-varying polynomial operator, and derive a number of module-theoretical results to be used in latter sections. It is known that if a system in state-space form is uniformly observable, it has a fractional representation with monic coefficient [5]. In particular, given a pair of polynomial operators $(Q(t,p),R(t,p))$, with Q monic, we show how to define two (additive) time-invariant maps F and G over right modules, which form a formal system over a noncommutative ring. These are an extension to time-varying systems of some results of Fuhrmann [8]. These results of this section are equally valid for both discrete-time and continuous-time systems. Then, we also show how the monicity constraint can be removed again using these results.

In Section 3, we consider the (time-varying) polynomial equations

$$Q(t,p)Q_c(t,p) + R(t,p)P_c(t,p) = \phi(t,p),$$

$$\text{and } Q(t,p)X(t,p) + X(t,p)R(t,p) = \Phi(t,p)$$

where Q, R, ϕ are given polynomial operators with Q monic and Q_c, P_c,X,Y are polynomial operators to be found. We show that the solutions of these equations are characterized by solutions of the linear equations of the form

$$[G:FG:..:F^rG; \quad \begin{bmatrix} P_o \\ \vdots \\ P_r \end{bmatrix} = \hat{\phi}$$

where F, G are as defined in the Section 2, and $\hat{\phi}$ is a representation of $\phi(t,p)$ obtained using the results of Section 2. This follows the approach in Emre [25] and Emre and Silverman [21] very closely. The first polynomial equation and its solution can be used to obtain conditions for the existence (and constructions) of causal dynamic output feedback compensators which (algebraically) internally stabilize the closed-loop system (in terms of the reachability map obtained from (F,G)). This leads to algebraic stabilizability conditions for linear systems described as

$$Q(t,p)y(t) = R(t,p)u(t)$$

where Q is monic. (Also, the case $Q_1(t,p)\hat{x}(t) = u(t)$; $y(t) = P_1(t,p)\hat{x}(t)$ can be treated similarly using the duality results in Section 2.)

Also, detectability conditions (given in Emre [11]) also correspond to solving a (dual) polynomial equation, which can be also studied the same (dual) way. These results follow very closely the results in [15-18] for linear systems over rings. Furthermore, these results are applicable the same way to both continuous-time and discrete-time systems. Also, for continuous-time systems over the ring of real-valued analytic functions, the monicity constraint on $Q(t,p)$ may be weakened using a technique similar to a technique given in Emre [17].

The internal stability of the observer and dynamic state-feedback configuration is very closely related to these algebraic detectability and stabilizability conditions.

The state-space realizations for the required controllers and observers (obtained first in fractional form) fore regulation can be easily obtained.

For discrete-time systems situation is much more similar to the time-invariant case. Using formal power series, which are readily well defined in this case, one can obtain a realization of a fractional description ($P Q^{-1}R$) similar to the realization of Fuhrmann [24] (see [20], [14]). Finally, we show here how the skew time-varying polynomial equation can be similarly solved suing the results developed here, and following the approach in Emre and Silverman [21]. This equation is a key to the solution of output regulation/trucking problems (see [22-23]).

Some of our results are related to the results of Kamen and Khargonekar [15] where a dynamic state feedback result has been established for uniformly reachable discrete-time systems. This result can be obtained as a very special case of our Theorem 3.9 which establishes a general stabilizability result. Also, our results are more direct and do not require augmentation by integrators to obtain an index invariant system as in [19]. Furthermore, we also give the corresponding stabilizability result for continuous-time systems whereas only discrete-time systems are considered in [19]. After the first version of this paper was written, the author has received reference [14] from its authors. Several results for discrete-time case including stabilizability are given in [14] developed independently of our work. It is clear from the results of [20] and [14] that the discrete-time case can be extended in a way very similar to the case of linear time-invariant systems over fields. However, the results established here and [20] are also applicable to continuous-time case which is considerably more involved than the discrete-time case. For more details of some of the results reported here, we refer to [20].

2. PRELIMINARIES

In this section we will set up a framework upon which an algebraic theory of regulation of linear time-varying systems (discrete-time or continuous-time) by both state-feedback and observers, and by direct dynamic output feedback can be based. As it will be seen, the theory is very similar to the corresponding theory established in [15-18] for linear systems over rings.

Let p denote the left shift operator or the derivative operator. A (time-varying) polynomial operator in p is an operator of the form

$$(2.1) \qquad L(t,p) = \sum_{j=0}^{k} A_j p^j$$

where A_j are $\bar{p}xr$ matrices over either real-valued functions over integers (or a subring of it), or real-valued infinitely differentiable functions on real numbers. The ring of such functions will be denoted by k in either case. We assume $A_k \neq 0$. Thus, the polynomial operator L(t,p) represents a map

$$L(t,p): K^r \to K^p; \; x \to \sum_{j=0}^{k} A_j \, p^j x.$$

L(t,p) is called monic iff $A_k = I_{\bar{p}}$ (=identity matrix of size $\bar{p}$). The degree of an L(t,p) as in (2.1) is defined to be k, and denoted as $\delta(L)$. We define the product of a $\bar{p}xk$ operator $L_1(t,p)$ with a kxr operator $L_2(t,p)$ as in their composition as operators, $L_1(t,p)oL_2(t,p)$, and denote it as

$L_1(t,p)L_2(t,p)$. Next we present a simple (but important) observation for monic polynomial operators $L(t,p)$.

2.2. LEMMA. *Let* $Q(t,p)$ *be a* $\bar{p}x\bar{p}$ *operator which is monic. Let* $\phi(t,p)$ *be any* $\bar{p}xr$ *polynomial operator. Then there exist unique polynomial operators* $\phi_1(t,p)$ *and* $\Phi(t,p)$ *such that*

$$\phi(t,p) = Q(t,p)\phi_1(t,p) + \Phi(t,p), \tag{2.3}$$

and

$$\delta(\Phi) < \delta(Q).$$

PROOF. By simply equating the coefficients of both sides in (2.3) and using well known identities for terms of the form

$$B_1(t)p^{k_1}B_2(t)p^{k_2}$$

to express them as in (2.1). □

Note that Lemma 2.2 shows that left division of an operator $\phi(t,p)$ with a monic operator $Q(t,p)$ is well defined. We define the remainder of $\phi(t,p)$ to be $\Phi(t,p)$ and write this as

$$\Phi(t,p) = \phi(t,p) \bmod Q(t,p)$$

Based on Lemma 2.2, we can associate a projection π_Q with any given monic $\bar{p}x\bar{p}$ $Q(t,p)$ as follows.

Let $K^{\bar{p}}[p]$ denote the set of $\bar{p}x1$ polynomial operators as in (2.1). Let K_Q denote the subset of $K^{\bar{p}}[p]$ such that degrees of the elements of K_Q are less than degree of Q. Then

$$\pi_Q : K^{\bar{p}}[p] \to K_Q;$$

$$x(t,p) \mapsto x(t,p) \bmod Q(t,p).$$

One can define a right $K[p]$-module structure on $K^{\bar{p}}[p]$ in the usual way, i.e.,

$x(t,p)\cdot\alpha(t,p)$:=composition of the operators $U(t,p)\varepsilon K^{\bar{p}}[p]$ and $\alpha(t,p)\varepsilon K[p]$. Thus $K^{\bar{p}}[p]$ is a free right $K[p]$ module of rank $\bar{p}$.

One can also define a right $K[p]$-module structure on K_Q by

$$x(t,p)\alpha(t,):=\pi_Q(x(t,p)\cdot\alpha(t,p))$$

where the product inside π_Q ( ) is the usual operator composition. Clearly, a set of generators for K_Q are

$$\bar{\varepsilon}_i : = \pi_Q(\varepsilon_i)$$

where ε_i is the i-th unit vector in $K^{\bar{p}}$.

Note that both, $K^{\bar{p}}[z]$ and K_Q are also right K-modules where scalar multiplication is defined as above by restricting $\alpha(t,p)$ to $K^{\bar{p}}$. Next, we define three important maps which will be fundamental for our latter results.

Let $Q(t,p)$ be a pxp monic polynomial operator, and let $R(t,p)$ be any $\bar{p}xm$ polynomial operator. We define the maps

$$G : K^m \to K_Q$$

and

$$F : K_Q \to K_Q$$

as follows.

$$Gu : = \pi_Q(R(t,p)u), \text{ for } u\varepsilon K^p,$$

and

$$Fx : = \pi_Q(x\cdot p), \text{ for } x\varepsilon K_Q,$$

where the products inside brackets are again operator compositions.

With these definitions, one can easily verify that for any $u\varepsilon K^{\bar{p}}$ and any integer $i\geq 0$, we have

$$\pi_Q(R(t,p)up^i) = F^iGu.$$

Thus for any

$$u(t,p) = \sum_{u=0}^{k} u_j p^j,$$

we have

$$\pi_Q(R(t,p)u(t,p)) = \sum_{j=0}^{k} F^j G u_j .$$

Hence, with any given R(t,p) and monic Q(t,p), one can associate the above right K[p]-module homomorphism

$$K^{\bar{p}}[p] \to K_Q$$

compatible with the right module structures on $K^{\bar{p}}[p]$ and K_Q defined before. Indeed, for any fixed α(t,p) in K[p], we have

$$\pi_Q(R(t,p)u(t,p)\alpha(t,p))$$

$$= \pi_Q(\pi_Q(R(t,p)u(t,p)\alpha(t,p)).$$

This can be easily verified by remembering the definition of π_Q.

Also note that if

$$\phi(t,p) = \sum_{j=0}^{k} \phi_j p^j, \text{ we have}$$

$$\bar{\Phi} = \pi_Q(\phi) = \sum_{j=0}^{k} F^j \pi_Q(\phi_j). \quad \text{If } \delta Q \leq 1, \text{ then}$$

$$\pi_Q(\phi) = \sum_{j=0}^{k} F^j \phi_j .$$

2.4. REMARK. One can dualize the results established in this section easily. Again, if Q(t,p) is a monic polynomial operator, we can express any polynomial operator ϕ(t,p) uniquely as

$$\phi(t,p) = \phi_1(t,p)Q(t,p) + \bar{\Phi}(t,p),$$

with $\delta(\bar{\Phi}) < \delta\,(Q)$. Thus right-division by Q is possible. Then one can dualize all the rest of the results in terms of left module structures, left module homomorphisms, this time denoting vectors as row vectors. For example, then we will have

$$\pi_Q(u(t,p)R(t,p)) = \sum_{j=0}^{k} F^{T^j} H^T u_j$$

where

$$H^T : K^m \to K_Q \;;\; u \mapsto \pi_Q(uR(t,p))$$

$$F^T : K_Q \to K_Q \;;\; x \mapsto \pi_Q(p\cdot x).$$

Everything translates with the representation of vectors as rows, and replacing the word "right" by "left," G by H^T, and F by F^T.

Above for clarity of exposition, we constrained our results to the case where Q(t,p) is monic. Next, we show how these results can be extended to the general case.

2.5. DEFINITION. A $\bar{p}$x$\bar{p}$ polynomial operator Q(t,p) is called *left admissible* iff there exists a $\bar{p}$x$\bar{p}$ polynomial operator M(t,p) such that

(2.6) $\quad Q(t,p) := M(t,p)Q(t,p)$

is a monic operator.

Note that such an M(t,p) may not be unique, but this does not cause a difficulty as far as our results are concerned. Let Q(t,p) be admissible. From now on, we will assume throughout that an M(t,p) as in (2.6) is fixed. We define K_Q, π_Q as follows. Let $x(t,p) \varepsilon K^{\bar{p}}[p]$. Then by Lemma 2.2, there exist unique polynomial operators $x_1(t,p)$ and $\bar{x}(t,p)$ such that $\delta\bar{x} < \delta Q$, and

$$M(t,p)x(t,p) = Q(t,p)x_1(t,p) + \bar{x}(t,p).$$

We define

$$\pi_Q(x(t,p)) := x(t,p) - Q(t,p)x_1(t,p).$$

Define K_Q as

$$K_Q := \{\tilde{x}(t,p)\varepsilon K^{\bar{p}}[p] \mid \tilde{x}(t,p) = \pi_Q(x(t,p)) \text{ for some } x(t,p)\varepsilon K^{\bar{p}}[p]\}.$$

Thus

$$\pi_Q : K^{\bar{p}}[p] \rightarrow K_Q$$

is a well defined function which is a projection. Define a right $K[p]$-module structure on K_Q by

$$x(t,p)\cdot\alpha(t,p) := \pi_Q(x(t,p)\cdot\alpha(t,p)) \text{ for } x(t,p)\varepsilon K^{\bar{p}}[p],\ \alpha(t,p)\varepsilon K[p].$$

Considering the identity

$$\pi_Q(\pi_Q(x(t,p)\alpha(t,p)\beta(t,\beta)) = \pi_Q(x(t,p)\alpha(t,p)\beta(t,p)),$$

it can be seen that K_Q is a well defined right $K[p]$-module and

$$\pi_Q : K^{\bar{p}}[p] \rightarrow K_Q$$

is a right $K[p]$-module homomorphism whose kernel is $QK^{\bar{p}}[p]$. Next, define the maps F and G as before using the present definition of π_Q and K_Q. Then, these maps have all the properties mentioned above for a monic $Q(t,p)$.

We can also dualize these results for right-admissible operators $Q(t,p)$ and the properties mentioned in Remark 2.4 are still valid.

3. THE POLYNOMIAL EQUATION $QQ_c+RP_c = \phi$ AND REGULATION

First we will consider the polynomial equation

$$QQ_c + RP_c = \phi \tag{3.1}$$

where Q, R, ϕ ae given $\bar{p}x\bar{p}$, $\bar{p}xm$, and $\bar{p}xr$ (time-varying) polynomial operators, with Q left admissible. Q_c, P_c are time-varying polynomial operators to be found. We have the following translation of Emre [1, Theorem 2.3]. Define F, G as before.

3.2. THEOREM. *There exist polynomial operators* Q_c, P_c *satisfying* (3.1) *iff*

$$[G:FG:\ldots:F^rG]\begin{bmatrix} x_o \\ \vdots \\ x_r \end{bmatrix} = \pi_Q(\phi) \tag{3.3}$$

for some integer r *and for some matrices* x_i *over* K. *Furthermore, if* (3.1) *has a solution* Q_c, *and*

$$P_c = \sum_{j=0}^{r} P_jp^j, \tag{3.4}$$

then P_j *are given as solutions of the linear equations*

$$[G:FG:\ldots:F^rG]\begin{bmatrix} P_o \\ \vdots \\ P_r \end{bmatrix} = \pi_Q(\phi). \tag{3.5}$$

Next we will consider regulation of a linear time-varying system described as

$$Q(t,p)y(t) = R(t,p)u(t) \tag{3.6}$$

where $Q(t,p)$ is monic (Note that if a system in state-space form is uniformly observable, such a representation always exists (see [5])). We will assume that $\delta(R(<\delta(Q)$ (although this is not absolutely necessary for applicability of our results mathematically). Suppose, to the system described in (3.6), we apply a dynamic feedback system described as

$$Q_c(t,p)\hat{x}(t) = y(t)$$

(3.7)

$$u(t) = -P_c(t,p)\hat{x}(t) + v(t)$$

where $Q_c(t,p)$ is monic and $\delta(P_c) \le \delta(Q_c)$. $v(t)$ is an external input. The closed-loop sytem is described as

$$(QQ_c + RP_c)\hat{x} = Rv$$

$$y = Q_c\hat{x}.$$

Note that $QQ_c + RP_c$ is monic with degree $= \delta(QQ_c)$.

We will call such a system *algebraically internally stable* iff $\hat{x}$ and $p\hat{x},\dots,p^r\hat{x}$ where $r = \delta(QQ_c)-1$ approach zero exponentially with time for every initial conditions on $\hat{x},\dots,p^r\hat{x}$. Thus algebraic stability is determined by the polynomial operator.

$$QQ_c + RP_c.$$

we will call (Q,R) *algebraically stabilizable* iff there exist $\bar{p}x\bar{p}$, $\bar{p}x\bar{p}$ and $mx\bar{p}$ polynomial operators X_1, X_2, K with K monic such that

i) $QX_1 + RX_2 = K$,

ii) $(\delta K) \ge \delta(Q) + \delta(X_2)$, and

iii) all the solutions $\hat{x}$ of the equation

$$K(t,p)\hat{x}(t) = 0$$

have the property that $\hat{x},\dots,p^k\hat{x}$ ($k = \delta(K) - 1$) approach zero with time exponentially.

The next theorem states that a system as in (3.6) can be algebraically internally stabilized by causal dynamic feedback as in (3.7) iff (Q,R) is algebraically stabilizable.

3.8. THEOREM. *There exists a causal dynamic system as in* (3.7) *which (algebraically) internally stabilizes* (3.6) *iff* (Q,R) *is algebraically stabilizable*.

Next, we will apply these results to observerand dynamic state-feedback stabilization of linear sytems. In case of dynamic state-feedback, (3.6) becomes

$$(pI - F(t))x(t) = \tilde{G}(t)u(t)$$

In this case one can obtain representations of the maps F and G associated with $Q(t,p) = pI-\tilde{F}(t)$, $R(t,p) = \tilde{G}(t)$ as follows.

G becomes multiplication by $\tilde{G}(t)$.

F has different representations in discrete-time and continuous time cases. We have

$$xp = (pI-\tilde{F})Q_1 + Fx$$

Equating coefficients gives:

i) $F = \tilde{F}p^{-1}$, in the discrete-time case (where p^{-1} is the right shift operator).

ii) $F = -(pI-\tilde{F})$, in the continuous time case.

For any

$$\phi(t,p) = \sum_{j=0}^{r+1} \phi_j(t)p^j,$$

we have

$$\pi_{(pI-\tilde{F})}(\phi) = \sum_{j=0}^{r+1} F^j\phi_j.$$

Thus, in this case, for discrete-time systems we obtain the following Theorems concerning algebraic stabilizability of $(\tilde{F}(t), \tilde{G}(t))$:

3.9. THEOREM. *There exist time-varying polynomial operators*

$$Q_c(t,p) = p^{k+1}I + \sum_{j=0}^{k} Q_{cj}(t)p^j$$

$$P_c(t,p) = \sum_{j=0}^{k+1} P_{cj}(t)p^j$$

such that

$$(pI-\tilde{F})Q_c + \tilde{G}P_c = \phi$$

where

$$\phi(t,p) = p^{r+1}I + \sum_{j=0}^{r} \phi_j(t)p^j$$

iff there exists an integer $k < r$ *and time varying matrices* $P_{ci}(t)$; $0 \le i \le k+1$ *such that*

$$(3.10) \qquad [\tilde{G}(t):\tilde{F}(t)\tilde{G}(t-1):\ldots:\tilde{F}(t)\ldots.\tilde{F}(t-k)\tilde{G}(t-k-1)] \begin{bmatrix} \bar{P}_{co}(t) \\ \vdots \\ \bar{P}_{ck+1}(t) \end{bmatrix}$$

$$= \phi_o(t) + \tilde{F}(t)\phi_1(t-1)+\ldots+\tilde{F}(t)\ldots\tilde{F}(t-r)$$

(in which case $P_{ci} = p^i\underline{P}_{ci}$ for $0 \le i \le k+1$).

3.11. REMARK. For continuous-time case, the continuous-time version of Theorem 3.9 is the same except that (3.10) is replaced by the algebraic linear equations

$$(3.14) \qquad [\tilde{G}(t):(\tilde{F}(t)-pI)\tilde{G}(t):\ldots:(\tilde{F}(t)-pI)^{k+1}\tilde{G}(t)] \begin{bmatrix} \bar{P}_o(t) \\ \vdots \\ \bar{P}_{k+1}(t) \end{bmatrix}$$

$$= \phi_o(t) + (\tilde{F}(t)-pI)\phi_1(t)+\ldots+(\tilde{F}(t)-pI)^{r+1}\cdot I,$$

where P_i are required to be a sufficient number of differentiable functions: If (3.14) has solutions, then these solutions can be obtained by solving linear algebraic equations overthe function ring.

Next we will show that the results established in Section 2can also be used for the solution of the skew polynomial equation

(3.16) $\quad QY + XR = \Phi$

where $Q(t,p)$, $X(t,p)$, $Y(t,p)$, $R(t,p)$, $\Phi(t,p)$ are time-varying polynomial operators with Q, R, Φ given; X, Y unknown. We will assume that $Q(t,p)$ is left admissible and $K_Q$is a free right K-module. In particular, this is true if Q is monic for the continuous time case. It is reported in [14] that if Q has atwo sided inverse, this is always true for the discrete-time case. Here we will consider the continuous-time case, since the discrete-time case is essentially the same. The following brief characterization is based on the results on Section 2 and follows the approach in Emre and Silverman [21]. Clearly, if (3.16) is satisfied, we must have

$$\pi_Q(XR) = \pi_Q(\Phi), \tag{3.17}$$

But, by [21, Lemma 2.2], if (3.16) has a solution, it has one with columns of X is K_Q. Let S be a matrix whose columns are a basis of X. Then

$$X = S\hat{G}, \ \rho_Q(\Phi) = S\hat{\Phi}$$

for some matrices $\hat{G}, \hat{\Phi}$ over K. Let R(t,p) be expressed as

$$R(t,p) = \sum_{j=0}^{r} u_{-j}p^j$$

with u_j matrices over K. Thus (3.17) implies

$$\sum_{j=0}^{r} F^j S\hat{G}\, u_{-j} = S\hat{\Phi}. \tag{3.18}$$

Now, for any $x(t,p)\varepsilon K_Q$, $x(t,p) = Sx$ for some vector x over K. Then

$$F(x(t,p)) := \pi_Q(Sxp) = \pi_Q(Spx - S(px))$$

$$= \pi_Q(Sp)x - \pi_Q(S(px)).$$

Let $\hat{F}$ be the unique square matrix over K such that

$$F(S) = S\hat{F}.$$

Thus

$$F(x(t,p)) = S(\hat{F} - pI)x.$$

Thus (3.18) is yields

$$\sum_{j=0}^{r} (\hat{F}-pI)^j \hat{G} u_{-j} = \hat{\Phi}. \tag{3.19}$$

Conversely, if $\hat{G}$ satisfies (3.19), then it satisfies (3.18) which implies (3.17), i.e., x =$S\hat{G}$ isa solution of (3.16). The corresponding Y can be obtained similarly. Thus the solution of (3.16) is equivalent to the solution of the linear differential equation in $\hat{G}$ given by (3.19). The dual equation

$$YQ + RX = \Phi,$$

with Q right admissible, can be solved similarly. (3.16) is a key equation in the solution of output regulation/trucking problems, together with the results developed in [22] which applies to differential operators. Thus,the results here, together with the results in [22] provide considerable basis for the approach to the output regulation/tracking developed in [23] to be applicable in the time-varying case.

8. REFERENCES

[1] R. E. Kalman, "Contributions to the theory of optimal control," Boletin De La SociedadMatematica Mexicana, pp. 102-119, 1960.

[2] R. E. Kalman and R. S. Bucy, "New results in linear filtering and prediction theory," Trans. ASME, J. Basic Eng., Ser. D., Vol. 83, pp. 95-108, March 1981.

[3] L. M. Silverman and H. E. Meadiwsm :Controllability and observability in time-variable linear systems," SIAM J. Contr., 5, 64-73, 967.

[4] A. S. Morese and L. M. Silverman, "Structure of index-invariant systems," SIAM J. Contr., Vol. 11, No. 2, May 1973.

[5] L. M. Silverman, "Discrete Riccati equations: alternative algorithms, asymptotic properties, and system theoretic interpretations," in Control and Dynamical Systems, Vol. 12, (Ed. by C. T. Leondes), Acad. Press, 1976.

[6] H. D'angelo, Linear Time-Varying Systems: Analysis and Synthesis, Allyn and Bacon, Boston, 1970.

[7] R. E. Kalman, P. L. Falb and M. A. Arbib, Topics in Mathematical System Theory, McGraw Hill, 1969.

[8] L. M. Silverman, "realization of linear dynamical systems," IEEE Trans. Autom. Contr., Vol. AC-16, pp. 554-567, 1971.

[9] E. W. Kamen, "New results in realization theory for linear time-varying analytic systems," IEEE Trans. Autom. Contr., Vol. AC-24, No. 6, Dec. 1979.

[10] E. W. Kamen and K. M. Hafez, "Algebraic theory of lineartime-varying systems," SIAMJ. Contr. and Opt., Vol. 17, pp. 500-510, 1979.

[11] E. Emre, "Observer for nonlinear/time-varying sytem," Proc. 21st Allerton Conference on.

[12] R. Ylinen, "On the algebraic theory of linear differential and difference systems with time-varying or operator coefficients," Ph.D. Thesis, Helsinki Univ. of Technology, Finland, 1975.

[14] E. W. Kamen, P. P. Khargonekar, and K. R. Poola, "Representation and control of linear time-varying systems and polynomial-matrix fractions," Preprint, (MST, Univ. Flo., March 1983.

[15] E. Emre, "The polynomial equation $QQ_c + RP_c = \phi$ with application to dynamic feedback," SIAM J. Contr. and Opt., Nov. 1980.

[16] ________, "On necessary and sufficient conditions for regulation of linear systems ovr rings," SIAM J. Contr. and Opt., March, 1982.

[17] ________, "Regulation of linearsystems over rings by dynamic output feedback," Systems and Control Letters, 3;57-62, 1983.

[18] E. Emre and P. P. Khargonkar, "Regulation of spitlinear systems ovr rings: coefficient-assignment and observers," IEEE Trans. Autom. Contr., March 1982.

[19] E. W. Kamen and P. P. Khargonekar, "A transfer function approach to linear time-varying discrete-time systems," Proc. IEEE Decis. andContr., 1982.

[20] E. Emre,"A polynomial fractionalapproach to linear time-varying systems," Peprint, Dept. of Elec. Eng./Comp. Sci., Texa Tech University, February 1983.

[21] E. Emre and L. M. Silverman, "The equation $XR + QY = \phi$: A characterzaton of solutions,:" SIAM J. Contr. and Opt., Jan. 1981.

[22] E. Emre, "further results on skiw prime matrices," Proc. IEEE Conf. on Decis. and Contr., 1982.

[23] E. Emre, A. B. Ozgulaer and P. P. Khargonekar, "Output regulation nad tracking of linear systems over rings," Proc. IEEE Conf. on Decis. and Contr., 1982.

[24] P. A. Fuhrmann, "Algebraic system theory: An analyst's point of view," J. Franklin Inst., Vol. 301, 6, June 1976.

COEFFICIENT ASSIGNMENT FOR SISO TIME DELAY SYSTEMS†

John Chiasson
Center for Control Sciences and Dynamical Systems
University of Minnesota

SUMMARY*

In this paper the problem of coefficient assignment for SISO time delay systems is considered. We use an input-output representation of the form $h(s,z) = b(s,z)/a(s,z)$ where $z = e^{-hs}$, $h > 0$ is the delay duration, $a(s,z) \varepsilon R[s,z]$ and monic in s, $b(s,z) \varepsilon R[s,z]$ and $\deg_s b(s,z) < \deg_s a(s,z)$. We approach the problem in the frequency domain by using a Wolovich type compensator [1] to do coefficient assignment.

Lévy [2] has considered this problem using the same aforementioned compensator structure and has even considered the MIMO case. Here we look at the problem in greater depth (for the SISO case) to show that the whole set of compensators (rather their characteristic polynomials) for a given system $h(s,z)$ can be generated from a finite "basis". All (and only) compensators from this set will work to achieve arbitrary coefficient assignment, stability withstanding. Thus a designer is left with choosing a characteristic polynomial from this set which has desirable stability characteristics. In other words, we show that in order to achieve arbitrary coefficient assignment, the characteristic polynomial of the compensator must be given by (and only by) a two variable polynomial of the form

$$\delta(s,z) = a(s,z) + \sum_{0}^{n-1} p_i(z)\gamma_i(s,z)$$

where $a(s,z)$ is the characteristic polynomial of the original system,

†This research has been supported by a Dissertation Fellowship from the Graduate School of the University of Minnesota

* For the complete paper write the author at the following address:

University of Minnesota
139 Electrical Engineering
123 Church Street, S.E.
Minneapolis, MN 55455

$n \triangleq \deg_s a(s,z)$, $p_i(z) \in R[z]$, and $\gamma_i(s,z) \in R[s,z]$ with $\deg_s \gamma_i(s,z) \leq n-1$ for $i = 0, \ldots, n-1$ is a "compensator basis" determined simply and explicitly be $a(s,z)$ and $b(s,z)$. The $p_i(z)$ will later be subject to the (troublesome) constraint that $\delta(s,z)$ be stable.

Emre and Khargonekar [3] also consider dynamic compensation but their procedure entails finding a solution to a matrix equation (even in the SISO case) under the constraint that the characteristic polynomial of the observer is stable. It appears that the method presented here (at least for the SISO case) would eliminate this "iterative" procedure. Again, this would be accomplished by finding the whole set of "admissable" compensators as noted above and then choosing a stable one. Regulation of linear systems over rings (e.g. time delay systems) by dynamic output feedback has also been considered by Emre in [4].

Summarizing, we give an explicit characterization of coefficient assignment (stability withstanding) for SISO time delay systems. We illustrate the technique with a few examples and then give a characterization of reduced order compensators.

References

[1] W. L. Wolovich, Linear Multivariable Systems, Springer-Verlag, 1979.

[2] B. C. Lévy, 2-D polynomial and rational matrices, and their applications for the modeling of 2-D dynamical systems, Technical Report No. M735-11, Stanford Univ., June 1981.

[3] E. Emre and P. P. Khargonekar, Regulation of Split Linear Systems over Rings: Coefficient-Assignment and Observers, IEEE Trans. on Auto. Contr., Vol. AC-27, February 1982.

[4] E. Emre, Regulation of Linear Systems over Rings by a Dynamic Output Feedback, Systems and Control Letters, June, 1983.

EXISTENCE OF DIAGONAL CONTROLLERS FOR A CLASS OF LINEAR TIME-INVARIANT SYSTEMS

K. NATARAJAN and R. BALASUBRAMANIAN
Department of Electrical Engineering
University of New Brunswick, Fredericton, Canada

ABSTRACT

In this paper, the algebraic characterization of fixed modes [1] is used to develop a sufficiency condition for the existence of diagonal stabilizers for unity feedback systems when the plant transfer function is allowed to be modified by a pre/post compensator.

1. THE MAIN RESULT

1.1 Notation

T is a polynomial matrix postcompensator around a plant whose transfer function is G and C is the transfer function of the diagonal stabilizer for the resulting unity feedback system. G is an nxm ($n \geq m$) transfer matrix of rational functions. ϕ_G is the characteristic polynomial of G. The feedback structure is shown schematically in Fig. 1.

1.2 Theorem

If any one column of $\phi_G G$ has its elements right half plane (rhp) coprime then a realizable rational postcompensator can be found so that the resultant transfer function can be stabilized by a diagonal controller.

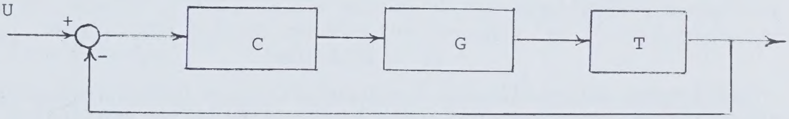


Fig. 1 Configuration of System

Proof:

We first find a T such that $|T|$ is a strictly Hurwitz polynomial. To make T realizable, it suffices to divide it by a strictly Hurwitz polynomial of appropriate degree.

The fixed modes of the unity feedback system with plant transfer function TG and a diagonal compensator are given by the zeroes of the polynomial [1]

$$\gamma = \mathrm{GCD}\{\phi_{TG}, p_1^1, p_2^2, p_m^m, p_{12}^{12}, p_{23}^{23}, \cdots \cdot p_{12..m}^{12..m}\} \tag{1}$$

where

$$p_{j_1 j_2 .. j_k}^{i_1 i_2 .. i_k} = \phi_{TG} \sum_{1 \leq l_1 \leq l_2 \leq l_k \leq n} T_{l_1 l_2 .. l_k}^{i_1 i_2 .. i_k} G_{j_1 j_2 .. j_k}^{l_1 l_2 .. l_k}$$

and T_{lm}^{ij}, G_{lm}^{ij} denotes the minor formed by the i,j rows and l,m columns of the matrix T,G respectively. Assume that the k^{th} column of G satisfies the condition of the theorem. There exists polynomials T_j^k such that

$$\phi_G \sum_{j=1}^{n} T_j^k G_k^j = \alpha \tag{2}$$

where α is the strictly Hurwitz GCD of $\{\phi_G G_k^j, j = 1,n\}$

Equation (2) also implies that the polynomials T_j^k, j = 1,n are also rhp coprime. If α_1 is their strictly Hurwitz GCD then there exists a T (for a constructive proof see [2]) with its kth row as

T_j^k , j = 1,n and $|T| = \alpha_1$.

With this T, ϕ_{TG} will have the same rhp factors as ϕ_G and from equation (2) p_k^k is strictly Hurwitz and the theorem is proved.

1.3 Corollaries

1. If any one element of $\phi_G G$ has no factors in the rhp, then the conditions of the above theorem are satisfied.
2. If any one row of $\phi_G G$ has its elements rhp coprime, then a realizable rational precompensator can be found so that the resultant transfer function can be stabilised by a diagonal controller.

1.4 Example

$$G = \begin{bmatrix} 1/s+4 & (s-1)/(s+4) \\ (s+2)/(s+4)(s-1) & 1/(s+4) \end{bmatrix}; \phi_G = (s+4)^2(s-1)$$

The (2,1) element of $\phi_G G$ has no zeroes in the rhp.

Hence, $T = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ is the requisite post compensator.

NOTE: Although in the proof we have used GCDs α, α_1, it suffices if α, α_1 are strictly Hurwitz.

2. CONCLUSION

The theorem and corollaries developed above are useful in finding pre/post compensators and from their forms having better qualitative understanding and quantitative feel for decentralised control of the system. Additionally the theorems may be useful in reordering inputs/outputs as in the above example.

REFERENCES

1. Vidyasagar, M. and Vishwanadham, N., 'Algebraic characterisation of decentralised fixed modes and pole assignment', Dept. of Elelctrical Engg., University of Waterloo report No. 82-06, Sept. 1982.
2. Newman, M., 'Integral matrices', Academic Press, 1972.
3. Natarajan, K., 'Design of linear multivariable feedback systems', M.Sc.E. Thesis, Dept. of Electrical Engg., University of New Brunswick, 1983.

MODULE CONSIDERATIONS FOR FEEDBACK SYNTHESIS OF SENSITIVITY COMPARISONS*

KENNETH P. DUDEK AND MICHAEL K. SAIN
Department of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

BOSTWICK F. WYMAN
Department of Mathematics
The Ohio State University
Columbus, Ohio 43210

ABSTRACT

This paper deals with the comparison of two general feedback control systems, both designed to achieve the same performances from plant control actions and plant output responses when the plant is at its nominal value. These nominal responses, M^0 and T^0, respectively, are described in terms of k(s)-linear maps $R(s) \to U(s)$ and $R(s) \to Y(s)$, where k is a field and R, U, and Y are finite dimensional k-vector spaces.

When the plant is not nominal, the two control systems produce instead the pairs (M_i, T_i), $i = 1,2$, which result in the command response errors $(T_i - T^0) = \Delta T_i$, $i = 1,2$. The comparison is posed in terms of the existence, and uniqueness, of comparison sensitivity morphisms (CSMs), such as $S : Y(s) \to Y(s)$ with $\Delta T_2 = S\Delta T_1$. Because control systems are designed to be "good", it is possible to speak of a good performance T^0, which can be restricted in the manner $T^0{}_g\colon\ R(s)_g \to Y(s)_g$, where g denotes intuitively a good transfer function and stands algebraically for ring localization. If the systems remain good under plant perturbation, then $(\Delta T_i)_g$, $i = 1,2$, can be formed; and there is the possibility of a CSM $\underline{S}$: im $(\Delta T_1)_g \to$ im $(\Delta T_2)_g$. $\underline{S}$ is a module-level idea, while S is a vector space idea. "In between" $\underline{S}$ and S is the option of yet another CSM, $S_g : Y(s)_g \to Y(s)_g$. Most of the details concern the passages "up" from $\underline{S}$ to S_g and "down" from S to S_g.

The study does not assume that either control system is open loop. Except for one result it does not assume that ΔT is epic. No particular controller implementation is assumed; and so the ideas generated pertain to general feedback synthesis. The zero module of ΔT_1 plays a central role; and the results seem to be of interest in H^∞ optimization.

INTRODUCTION

Let k be a field, and let R, U, and Y be finite dimensional k-vector spaces of plant output requests r, plant control inputs u, and plant outputs y. Let k(s) be the field of rational functions, with U(s), R(s), and Y(s) the k(s)-vector spaces corresponding to R, U, and Y according to the usual convention $R(s) = k(s) \otimes_k R$ and so forth. Consider systems identified with their command response morphisms $T_i : R(s) \to Y(s)$, $i = 1,2$, which under nominal plant conditions achieve the specified morphism T^0. Define the error morphisms $\Delta T_i = T_i - T^0$, $i = 1,2$. Intuitively, ΔT_1 and ΔT_2 represent perturbations of T^0 in two different feedback configurations, both of which are designed to control the same plant $P : U(s) \to Y(s)$, itself a morphism perturbed from its nominal character P^0. We are interested in comparing ΔT_2 with ΔT_1.

---

*Mr. Dudek's work was supported by the National Science Foundation (NSF) under Grant ECS-8102891; Dr. Sain's work was supported in part by the same NSF grant, and in part by the National Aeronautics and Space Administration (NASA) under Grant NAG 2-35; Dr. Wyman's work was supported by NASA under Grant NAG 2-34.

A $k(s)$-linear map $S : Y(s) \to Y(s)$ will be called a comparison sensitivity morphism (CSM) if $\Delta T_2 = S\Delta T_1$. Linear algebra immediately provides two facts concerning S. First, S exists if and only if $\ker \Delta T_1 \subset \ker \Delta T_2$. Second, S is unique if and only if $\operatorname{im} \Delta T_1 = Y(s)$. The idea of making such a comparison can be traced to the middle 1960s [1-3]; and the possibilities for applying it in design have been examined for nearly two decades. See, for example, [4-8]. By now, it seems to be reasonably common knowledge that design applicability of the comparison concept is influenced heavily by the configuration employed for controller implementation. When $\operatorname{im} \Delta T_1$ is rich, and when the controller configuration is decided, CSMs become virtually identified with essential control quantities, such as return differences and their inverses. Because of this, the CSM reaches in spirit back to the antiquities of control theory, while standing transparently in the background of ongoing modern investigations [9-14].

The motivations underlying this report are threefold: first, we wish to examine in greater detail the case in which ΔT_1 does not arise from an open loop system; second, we desire to learn more about the case in which $\operatorname{im} \Delta T_1$ is not rich; and third, we believe that the general theory of feedback synthesis requires a greater delineation of which features of CSMs can be known prior to acquiring knowledge of the controller implementation.

In this initial effort, we provide certain answers and raise some questions. A framework for the discourse is set up in the next section.

BASIC ISSUES

Let $k[s]$ be the domain of polynomials, and let $s_g \subset k[s]$ be a subset of nonzero "good" polynomials, closed under multiplication, and including the multiplicative unit. Denote by $k(s)_g$ the localized ring $s_g^{-1} k[s]$ of elements in $k(s)$ with "good" denominators. For any finite-dimensional vector space V over k, establish the notations $V[s] = k[s] \otimes_k V$ and $V(s)_g = k(s)_g \otimes_k V$ and observe that $V[s] \subset V(s)_g \subset V(s)$. A $k(s)$-linear map $T : R(s) \to Y(s)$ will be called a good morphism if $T(R(s)_g) \subset Y(s)_g$, in which case there is a restriction $R(s)_g \to Y(s)_g$ of T, which we denote by T_g. T_g is $k(s)_g$-linear.

With these conventions, we may ask if a CSM is good or not. Recall, now, that T^0 is certain to be good, by design. Moreover, it is common in current analyses of feedback around perturbed plants to make the assumption that the T_i are good also for $i = 1,2$. Clearly, then, ΔT_i, $i = 1,2$, are commonly assumed to be good. In such a situation, we may inquire about the existence and uniqueness of a good CSM, say S_g, such that $(\Delta T_2)_g = S_g (\Delta T_1)_g$. Whereas S was $k(s)$-linear, S_g would be $k(s)_g$-linear.

Two basic issues arise. Not surprisingly, we initially examine the first; and this has to do with how S_g relates to S. A natural way to pose the issue is by way of the commutative diagram of Figure 1, which assumes that S is given and inquires about S_g. Given that S exists on the

$$\begin{tikzcd}
Y(s)_g \arrow[r, "\text{inclusion}"] \arrow[d, dashed, "S_g"] & Y(s) \arrow[d, "S"] \\
Y(s)_g \arrow[r, "\text{inclusion}"] & Y(s)
\end{tikzcd}$$

Figure 1. Coming Down.

k(s)-linear, or vector space, level, can we restrict S or "come down" to the $k(s)_g$-linear, or good module, level? We shall see shortly that Figure 1 has an intuitively believable answer. Notice at this point, however, that the reverse question: given S_g, find if possible S: yields trivially by localizing up to the k(s) level. The second basic issue concerns what happens "below" the S_g level, so to speak. Here we begin by regarding ΔT_i, i = 1,2, as good morphisms with domain $R(s)_g$ and codomains $\text{im}\Delta T_i$, i = 1,2, respectively. Standard results from module theory [15, p. 26] can be applied to show that there exists a unique $\underline{S}: \text{im}\ (\Delta T_1)_g \rightarrow \text{im}\ (\Delta T_2)_g$ such that $(\Delta T_2)_g = \underline{S}(\Delta T_1)_g$ if and only if $\ker\ (\Delta T_1)_g \subset \ker\ (\Delta T_2)_g$, understood on the free $k(s)_g$-module level. The interesting question arises here when we ask if we can "go up" from $\underline{S}$ to S_g, a question which is pictured in Figure 2. This question occurs under favorable hypotheses, in

$$\begin{tikzcd}
\text{im}(\Delta T_1)_g \arrow[r, "\text{inclusion}"] \arrow[d, "\underline{S}"] & Y(s)_g \arrow[d, dashed, "S_g"] \\
\text{im}(\Delta T_2)_g \arrow[r, "\text{inclusion}"] & Y(s)_g
\end{tikzcd}$$

Figure 2. Going Up.

many ways, for $V(s)_g$ is a finitely generated free $k(s)_g$-module under our assumptions. Nonetheless, it is not necessarily true that $\text{im}\ (\Delta T_1)_g$ is a direct summand of $Y(s)_g$. As a result, "going up" can present nontrivial system theoretic questions.

The controlling factor associated with Figure 2 is the nature of the quotient module $Y(s)_g/\text{im}\ (\Delta T_1)_g$. Because $Y(s)_g$ is finitely generated, so is the quotient. Moreover, $k(s)_g$ is a principal ideal domain, and so the quotient can be written as the direct sum of a free part and a torsion part. If the torsion part vanishes, then we may "go up" to S_g in Figure 2, with S_g non-unique in close analogy to the vector space case. The presence of a non-zero torsion part is a very realistic possibility, however, and results in one of the "surprises" of the study.

KERNEL QUESTIONS

In the foregoing paragraphs, we spoke of the existence of S on the k(s)-vector space level and of $\underline{S}$ on the $k(s)_g$-module level. Whether we can "come down" from S to S_g or "go up" from $\underline{S}$ to S_g clearly depends fundamentally upon whether $\underline{S}$ and S themselves exist. The existence conditions were described, as usual, in terms of kernel inclusions. We begin this section with a result involving the relationship between two kernel inclusions.

Lemma. Let $T_i : R(s) \to Y(s)$, $i = 1,2$, be good k(s)-linear maps and let $(T_i)_g : R(s)_g \to Y(s)_g$ denote their restrictions to the $k(s)_g$-linear level. Then, $\ker T_1 \subset \ker T_2$ if and only if $\ker (T_1)_g \subset \ker (T_2)_g$.

Proof: For sufficiency, we assume that $\ker (T_1)_g \subset \ker (T_2)_g$. Consider $r(s) \in \ker T_1$. Thus r(s) has the representation $(n(s)/d(s)) \otimes_k r$ for n(s) and $d(s) \neq 0$ in k[s] and $r \in R$. It follows that $d(s)\, r(s) \in R(s)_g$, is an element of $\ker T_1$, is an element of $\ker (T_1)_g$ and thus of $\ker (T_2)_g \subset \ker T_2$; accordingly r(s) is an element of $\ker T_2$ as well. Necessity is more straightforward, since

$$\ker (T_1)_g = R(s)_g \cap \ker T_1 \subset R(s)_g \cap \ker T_2 = \ker (T_2)_g.$$

With the lemma in hand, existence of $\underline{S}$ and S can be established, provided that the two systems identified with T_1 and T_2 are designed to be nominally equivalent in the sense shortly to be described. We shall follow the path outlined in [16] for this demonstration.

We say that a k(s)-linear command response morphism $T^0 : R(s) \to Y(s)$ and a k(s)-linear control response morphism $M^0 : R(s) \to U(s)$ are a solution to the Nominal Design Problem (NDP) of linear multivariable control if they are good morphisms whose restrictions to $R(s)_g$ satisfy the commutative diagram of Figure 3, for some good morphism X_g. N_g and D_g are localizations

$$\begin{tikzcd}
 & & R(s) \arrow[ddddll, "M^0"'] \arrow[ddddrr, "T^0"] & & \\
 & & R(s)_g \arrow[u] \arrow[d, "X_g"] \arrow[ddl, "M^0_g"'] \arrow[ddr, "T^0_g"] & & \\
 & & U(s)_g \arrow[dl, "D_g"'] \arrow[dr, "N_g"] & & \\
 & U(s)_g \arrow[dl] & & Y(s)_g \arrow[dr] & \\
U(s) \arrow[rrrr, "P^0"] & & & & Y(s)
\end{tikzcd}$$

Figure 3.

of $N : U[s] \to Y[s]$ and $D : U[s] \to U[s]$, a right coprime factorization [17] of P^0 over $k[s]$.

Now consider the output feedback system of Figure 4. The controller $C : R(s) \oplus Y(s) \to U(s)$ is taken to be

```
                    * * * * *             * * * * *
                    *       *             *       *
      r → → → → → → *   C   * → → u → → *   P   * → → → → → → y
                    *       *             *       *     ↓
                    * * * * *             * * * * *     ↓
                        ↑                               ↓
                        ↑                               ↓
                        ↑ ← ← ← ← ← ← ← ← ← ← ← ← ← ← ← ←
```

Figure 4.

k(s)-linear, so that there exist unique k(s)-linear $C_Y : Y(s) \to U(s)$ and $C_R : R(s) \to U(s)$ with the property $C = C_R \pi_R + C_Y \pi_Y$ for π_R and π_Y the canonical projections out of the biproduct [18, p. 179]. C will be called an <u>output feedback synthesis</u> of the NDP solution (M^0,T^0) if $(1_{U(s)}-C_YP^0)^{-1} : U(s) \to U(s)$ exists and satisfies

$$M^0 = (1_{U(s)}-C_YP^0)^{-1}C_R.$$

<u>Theorem</u>. Let C_i, $i = 1,2$, be two output feedback syntheses of an NDP solution (M^0,T^0). Further, let T_i, $i = 1,2$, be the command response morphisms produced by the ith synthesis for the perturbed plant P. Then $\ker \Delta T_1 = \ker \Delta T_2$ as k(s)-vector spaces. Moreover, if T_i, $i = 1,2$, and T^0 are good morphisms, then $\ker (\Delta T_1)_g = \ker (\Delta T_2)_g$ as $k(s)_g$-modules.
Proof: Well known calculations can be employed to establish that

$$\Delta T_i = (1_{Y(s)}-P(C_i)_Y)^{-1}(P-P^0)(1_{U(s)}-(C_i)_YP^0)^{-1}(C_i)_R.$$

Since each C_i is an output feedback synthesis of (M^0,T^0), we can rewrite in the manner

$$\Delta T_i = (1_{Y(s)}-P(C_i)_Y)^{-1}(P-P^0)M^0,$$

from which it follows that

$$\ker \Delta T_1 = \ker \Delta T_2 = \ker (P-P^0)M^0,$$

as desired. In view of the Lemma, the $k(s)_g$-level assertion is a consequence.

Accordingly, CSMs $\underline{S}$ and S exist whenever we are comparing output feedback syntheses of the same nominal design (M^0,T^0), provided that T^0 is good and that T remains good under plant perturbation. Moreover, $\underline{S}$ can be seen to be an isomorphism of $k(s)_g$-modules. Though $S|\mathrm{im}\, \Delta T_1$ is monic, the same is not true of S in general; and neither is S required to be epic. So we see that S need not be pinpointed by return differences and their inverses, though that possibility is not excluded.

COKERNEL QUESTIONS

We shall see momentarily that the issue of "coming down" from S to S_g has an intuitively appealing approach. Before looking at that approach, however, we wish to examine the role played by the cokernel $Y(s)_g/\mathrm{im}(\Delta T_1)_g$ in deciding whether im $(\Delta T_1)_g$ is a direct summand of $Y(s)_g$. In order to do so, we consider the quotient construction $p : Y(s)_g \to Y(s)_g/\mathrm{im}(\Delta T_1)_g$. We know [18, p. 184] from linear algebra that, because p is epic, ker p = im $(\Delta T_1)_g$ will be a direct summand of $Y(s)_g$ if im p is free. We note also that $Y(s)_g$ is free by construction, that ker p is a submodule of a free $k(s)_g$-module, and that ker p is free then as well. Under what conditions is im p free?

Suppose M is a free module and $N \subset M$ a submodule. If M/N is free, then there exists an epimorphism of free modules $M \to M/N$ whose kernel is N. Thus it is necessary that N be the kernel of an epimorphism $M \to \underline{M}$ of free modules, if M/N is to be free. On the other hand, suppose that $f : M \to \underline{M}$ is an epimorphism of free modules with kernel N. Then $\underline{M}$ is isomorphic to M/N, and so M/N is free as well. We see, therefore, that the situation of interest revolves around the question of whether im $(\Delta T_1)_g$ is itself a kernel, of an epic $k(s)_g$-linear map on free module onto free module. We may see this question as an exact sequence:

$$R(s)_g \xrightarrow{(\Delta T_1)_g} Y(s)_g \dashrightarrow \text{Free } k(s)_g\text{-module} \to 0.$$

Because this sequence may not always be completed exactly, we face an important observation: epimorphisms of modules, even free modules, even over principal ideal domains, may display *different characters* in their images and kernels.

More can be said about the role of the cokernel in relation to S_g. Consider the commutative diagram of Figure 5, which has exact rows. The left square in Figure 5 represents the issue of "going up" from $\underline{S}$ to S_g, which we have taken to be achieved. It turns out that we can define S' as follows. Let $\underline{y}$ be in $Y(s)_g/\mathrm{im}\ (\Delta T_1)_g$, and let y in $Y(s)_g$ be such that $\underline{y} = p_1 y$. Establish S' $(\underline{y})$ to be $(p_2 S_g)(y)$; diagram chasing can be used to verify that S' is well established and that it is a morphism. In other words, if we are able to "go up" from $\underline{S}$ to S_g, then it is necessary that there exist a $k(s)_g$-linear $S' : Y(s)_g/\mathrm{im}\ (\Delta T_1)_g \to Y(s)_g/\mathrm{im}\ (\Delta T_2)_g$ which is, moreover, naturally related to S_g.

$$\begin{tikzcd}
0 \arrow[r] & \mathrm{im}\ (\Delta T_1)_g \arrow[r, "i_1"] \arrow[d, "\underline{S}"] & Y(s)_g \arrow[r, "p_1"] \arrow[d, "S_g"] & Y(s)_g/\mathrm{im}\ (\Delta T_1)_g \arrow[r] \arrow[d, dashed, "S'"] & 0 \\
0 \arrow[r] & \mathrm{im}\ (\Delta T_2)_g \arrow[r, "i_2"] & Y(s)_g \arrow[r, "p_2"] & Y(s)_g/\mathrm{im}\ (\Delta T_2)_g \arrow[r] & 0
\end{tikzcd}$$

Figure 5.

UPS AND DOWNS

In this section, we take a partial look at the basic issues of "going up" from $\underline{S}$ to S_g and "coming down" from S to S_g.

Going Up

We have said earlier that going up could be accomplished in a straightforward way, analogous to the vector space case, if $Y(s)_g/\mathrm{im}\,(\Delta T_1)_g$ is free. Here we illustrate what complications can occur in general. Denote $k(s)_g$ by A, and let $Y(s)_g = A^3$. Choose the standard ordered basis for $Y(s)_g$. Suppose that $\mathrm{im}\,(\Delta T_i)_g$, i = 1,2, is the span of the pair $\{(\varepsilon_1,0,0), (0,\varepsilon_2,0)\}$ where $\varepsilon_1/\varepsilon_2$ is not in A. For bases of $\mathrm{im}\,(\Delta T_i)_g$, i = 1,2, choose this pair, in the order indicated. We could then envision a case in which $\underline{S}$ would be a map with matrix

$$[\underline{S}] = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

Correspondingly, we could have

$$[i_1] = \begin{bmatrix} \varepsilon_1 & 0 \\ 0 & \varepsilon_2 \\ 0 & 0 \end{bmatrix} ; \quad [i_2] = \begin{bmatrix} \varepsilon_1 & 0 \\ 0 & \varepsilon_2 \\ 0 & 0 \end{bmatrix} .$$

If we could go up to S_g, then

$$[S_g][i_1] = [i_2][\underline{S}],$$

from which it follows that

$$[S_g] = \begin{bmatrix} 0 & \varepsilon_1/\varepsilon_2 & - \\ \varepsilon_2/\varepsilon_1 & 0 & - \\ 0 & 0 & - \end{bmatrix},$$

which means that $[S_g]$ has elements not in A. The meaning of $\varepsilon_1/\varepsilon_2$ not in A is that ε_2 has numerator factors not in s_g. As observed previously, we can overcome such difficulties if $Y(s)_g/\mathrm{im}\,(\Delta T_1)_g$ is free. Because $Y(s)_g$ is finitely generated, and because $k(s)_g$ is a principal ideal domain, we have that [17, p. 122]

$$Y(s)_g/\mathrm{im}\,(\Delta T_1)_g \sim \Big(\bigoplus_{i=1}^{t} k(s)_g/\varepsilon_i\, k(s)_g\Big) \oplus (k(s)_g)^{p-t}$$

where p is the dimension of Y, t is the rank of $\mathrm{im}\,(\Delta T_1)_g$, and ε_i is the ith invariant factor of $(\Delta T_1)_g$. Thus $Y(s)_g/\mathrm{im}\,(\Delta T_1)_g$ is free if and only if each ε_i is in s_g. These conditions obtain a convenient form in terms of the zero module $Z(\Delta T_1)$. From [17],

$$Z(\Delta T_1) = \frac{(\Delta T_1)^{-1}(Y[s]) + R[s]}{\ker \Delta T_1 + R[s]} .$$

As a finitely generated k[s]-torsion module, $Z(\Delta T_1)$ admits a minimal polynomial. The irreducible (over k[s]) factors of this polynomial are the support of $Z(\Delta T_1)$. We say the support is good if each element in the support is in s_g. Thus im $(\Delta T_1)_g$ is a direct summand of $Y(s)_g$ precisely when the support of $Z(\Delta T_1)$ is good, which is precisely when the transmission zeros of ΔT_1 are good.

Coming Down

Suppose given once again good morphisms T^0, T_1, and T_2. For simplicity in this subsection, we assume that im $\Delta T_1 = Y(s)$. Then S is uniquely determined. Suppose $y(s) \varepsilon Y(s)_g$, the basic issue is whether or not $Sy(s) \varepsilon Y(s)_g$. Consider the largest submodule of $Y(s)_g$ on which this is true, namely

$$S^{-1}(Y(s)_g) \cap Y(s)_g;$$

denote it by M, which is a submodule of $Y(s)_g$. We can "come down" exactly when $M = Y(s)_g$. To measure the failure of this condition, construct

$$X_g = Y(s)_g/M \ .$$

Consider for a moment $S : Y(s) \rightarrow Y(s)$ as the input/output function of a system. The pole module of S, that is, the state module of the minimal realization of S, is

$$X(S) = \frac{Y[s]}{S^{-1}(Y[s]) \cap Y[s]} \ .$$

Now X_g is just the localization [19] of X(S) with respect to the good denominator set s_g. Localization with respect to s_g kills exactly the good modes of X(S), leaving only the bad modes (if any). Therefore we can conclude that $X_g = (0)$ and that S_g exists if and only if X(S) is good, where good is defined in a manner analogous to that of the zero module above.

CONCLUSIONS

In this paper, we have considered certain module theoretic and vector space aspects of the idea of sensitivity comparisons. A principal goal of the work was to develop such comparisons in a coordinate free way, and to determine what features could be examined without incorporating the specific details of controller implementation. Such features can then be used constructively in general theories of feedback synthesis.

Using the idea of localization of rings, modules, and module maps, we have incorporated a broad notion of design "goodness", at least in regard to poles and zeros. Because of this, the discussion took place on two levels: the $k(s)_g$-module level and the k(s)-vector space level.

Two very basic issues arose. When the image of ΔT_1 was "small", there was the question of extending the "micro" CSM $\underline{S}$ to a more complete S_g. This turned out to depend in a rather subtle way upon the right-half plane zero structure of ΔT_i. If ΔT_i did have right-half plane zeros, we saw that $Y(s)_g/\text{im } (\Delta T_i)_g$ might have a nonzero torsion part. In case ΔT_i had both good zeros and good poles, then we could extend with no more difficulty than in the vector space case. On the other hand, when S was uniquely defined, there was the question of whether it was good. This question was settled by looking at its pole module X(S), when S was considered as the transfer function of a dynamical system.

The first of the issues above deals with the very important case of limited, structured perturbations in the plant, and with the corresponding results on command response. It is of control interest because of the fact that we wish to use no more control action than necessary to compensate for such events and to leave any excess control action available to address other issues, such as signal disturbances. The second of the issues relates directly to such topics as internal stability, if one system is open loop. If not, it relates to the stability of the compensator itself in the reference system. In either of these two cases, the goodness of S is important to the success of optimization schemes such as those reported in [12] and [20].

REFERENCES

[1] J.B. Cruz, Jr. and W.R. Perkins, "A New Approach to the Sensitivity Problem in Multivariable Feedback Design", IEEE Transactions on Automatic Control, Vol. 9, pp. 216-223, July 1964.

[2] W.R. Perkins and J.B. Cruz, Jr., "The Parameter Variation Problem in State Feedback Control Systems", Journal of Basic Engineering, Transactions of ASME, Series D, pp. 120-124, 1965.

[3] J.B. Cruz, Jr. and W.R. Perkins, "Criteria for System Sensitivity to Parameter Variations", Proceedings Third IFAC Congress, pp. 18c.1-18c.7, June 1966.

[4] E. Kreindler, "On the Definition and Application of the Sensitivity Function", Journal of the Franklin Institute, Vol. 285, No. 1, pp. 26-36, January 1968.

[5] M.K. Sain, A. Ma, and D. Perkins, "Sensitivity Issues in Decoupled Control System Design", Proceedings Twelfth Southeastern Symposium on System Theory, pp. 25-29, May 1980.

[6] M.K. Sain, R.M. Schafer, and K.P. Dudek, "An Application of Total Synthesis to Robust Coupled Design", Proceedings Allerton Conference on Communication, Control, and Computing, pp. 386-395, October 1980.

[7] M.K. Sain and A. Ma, "Multivariable Synthesis with Reduced Comparison Sensitivity", Proceedings Joint Automatic Control Conference, Vol. 1, paper WP8-B, 1980.

[8] M.K. Sain and J.L. Peczkowski, "An Approach to Robust Nonlinear Control Design", Proceedings Joint Automatic Control Conference, paper FA-3D, 1981.

[9] J.C. Doyle and G. Stein, "Multivariable Feedback Design: Concepts for a Classical/Modern Synthesis", IEEE Transactions on Automatic Control, Vol. AC-26, No. 1, pp. 4-16, February 1981.

[10] J.B. Cruz, Jr., J.S. Freudenberg, and D.P. Looze, "A Relationship between Sensitivity and Stability of Multivariable Feedback Systems", *IEEE Transactions on Automatic Control*, Vol. AC-26, No. 1, pp. 66-74, February 1981.

[11] M.G. Safonov, A.J. Laub, and G.L. Hartmann, "Feedback Properties of Multivariable Systems: The Role and Use of the Return Difference Matrix", *IEEE Transactions on Automatic Control*, Vol. AC-26, No. 1, pp. 47-65, February 1981.

[12] G. Zames, "Feedback and Optimal Sensitivity: Model Reference Transformations, Multiplicative Seminorms, and Approximate Inverses", *IEEE Transactions on Automatic Control*, Vol. AC-26, No. 2, pp. 301-320, April 1981.

[13] G. Zames and B.A. Francis, "A New Approach to Classical Frequency Methods: Feedback and Minimax Sensitivity", *Proceedings IEEE Conference on Decision and Control*, pp. 867-874, December 1981.

[14] G. Zames and B.A. Francis, "Feedback, Minimax Sensitivity, and Optimal Robustness", *IEEE Transactions on Automatic Control*, Vol. AC-28, No. 5, pp. 585-600, May 1983.

[15] T.S. Blyth, *Module Theory*. Oxford: The University Press, 1977.

[16] M.K. Sain, B.F. Wyman, R.R. Gejji, P.J. Antsaklis, and J.L. Peczkowski, "The Total Synthesis Problem of Linear Multivariable Control, Part I: Nominal Design", *Proceedings Joint Automatic Control Conference*, paper WP-4A, 1981.

[17] B.F. Wyman and M.K. Sain, "The Zero Module and Essential Inverse Systems", *IEEE Transactions on Circuits and Systems*, Vol. CAS-27, No. 2, pp. 112-126, February 1981.

[18] S. MacLane and G. Birkhoff, *Algebra*, Second Edition. New York: MacMillan Publishing Company, Inc., 1979.

[19] M.F. Atiyah and I.G. MacDonald, *Introduction to Commutative Algebra*. Reading, Mass.: Addison-Wesley, 1969.

[20] B.-C. Chang and J.B. Pearson, "Optimal Disturbance Reduction in Linear Multivariable Systems", Technical Report 8214, Electrical Engineering, Rice University, October 1982.

SWITCHED LINEAR SYSTEMS: STABILITY AND OBSERVER DESIGN

MARK MCINTYRE and M. VIDYASAGAR

Department of Electrical Engineering
University of Waterloo
Waterloo, Ontario

Abstract

A switched linear system is a discrete-time, finite-dimensional, real linear system in which the parameter variation is determined by the state of an autonomous linear system over a finite field. Necessary and sufficient conditions for switched linear system stability are given. Also, strong sufficient conditions which ensure that such systems are stable, regardless of the structure of the discrete-state system, are presented. It is shown that a switched linear system observer can be designed if both the continuous and discrete-state systems meet certain observability criteria.

Introduction

Switched linear systems (SLS's) are systems which can be modelled by the following state space equations:

$$x_{k+1} = A(j_k)x_k + B(j_k)u_k \tag{1}$$

$$y_k = C(j_k)x_k + D(j_k)u_k \tag{2}$$

$$j_{k+1} = Ej_k \tag{3}$$

$$l_k = Gj_k \tag{4}$$

We will refer to $x_k \in R^n$ as the continuous state of the SLS and to $j_k \in GF(p^l)^m$ as the discrete state. ($GF(p^l)^m$ denotes the vector space of ordered m-tuples over the finite field $GF(p^l)$.) The input and output of the continuous state system u_k, y_k each belong to a real vector space and the discrete system output l_k belongs to some finite vector space. $A(j_k), B(j_k), C(j_k)$ and $D(j_k)$ are real matrix-valued functions of j_k ,(of appropriate dimension), and E, G are constant matrices with elements from $GF(p^l)$. $k \in Z_+$ is a discrete-time index.

In order to characterize a particular SLS we need to know the matrices E, G and the functions $A(j_k), B(j_k), C(j_k)$ and $D(j_k)$. The information about the functions can be stated in terms of a mapping , called Γ, which identifies elements of the finite vector space with elements in finite sets of real matrices which we shall call $\{A\},\{B\},\{C\}$ and $\{D\}$.

$$\Gamma : GF(p^l)^m \to \{A\}\times\{B\}\times\{C\}\times\{D\} \tag{5}$$

In this paper we look at the problem of defining stability for SLS's and determining whether a specific SLS is stable. In particular, can we put conditions on the parameter matrix sets and the mapping Γ which ensure stability of a SLS? We also address the question of observer design for SLS's. Can we design an observer for a SLS to provide simultaneous estimates of the discrete and continuous states? Moreover, what conditions must a SLS satisfy to enable us to do this?

Before addressing these questions, we shall briefly consider the implications of the fact that the parameter variation in the continuous state system (equations (1) and (2)) is due to the evolution of the state of an autonomous linear system over a finite field.

Due to space limitations, the proofs of the theorems have been omitted. Proofs of the results are presented in [5].

Types of Parameter Variations

The state sequences generated by equation (3) for different initial discrete states j_0 are always periodic with a possible initial transient period [2],[3]. The number and period of these sequences depend on the Jordan structure of E [5]. This means that the parameter variation in the continuous state portion of a SLS is always eventually periodic.

Definition 1 A SLS is called *eventually periodic* if there is some initial discrete state which generates a discrete state sequence with period greater than 1 after a possible initial transient portion.

Definition 2 A SLS is called *eventually time-invariant* if all initial discrete states yield periodic sequences with period 1 after a possible initial transient portion.

We shall call the periodic systems which are realized for different initial discrete states the *realizable periodic systems* of a SLS.

Stability of SLS's

In the stability analysis of SLS's we will only consider internal stability; hence we deal with SLS's of the form

$$x_{k+1} = A(j_k)x_k \tag{6}$$

$$j_{k+1} = Ej_k \tag{7}$$

The definition of stability for SLS's is a natural one since the discrete state space has a finite number of elements and "unbounded" growth in the discrete state is not possible.

Definition 3 The SLS (6),(7) is *internally asymptotically stable* if, for all initial continuous states $x_0 \in R^n$ and for all initial discrete states $j_0 \in GF(p^l)^m$,

$$\lim_{k \to \infty} \| x_k \| = 0$$

where $\| \cdot \|$ is some norm on R^n.

Equivalently, we also say that 0 is an *asymptotically stable equilibrium point* of the SLS.

One approach to a stability analysis of SLS's is based on consideration of the stability of each realizable periodic system. It is not difficult to show that a necessary and sufficient condition for 0 to be an asymptotically stable equilibrium point of a T periodic , time-varying linear system is that

$$\rho(\Phi(T,0)) < 1 \tag{9}$$

where $\Phi(\cdot,\cdot)$ is the state transition matrix of the linear system and $\rho(\cdot)$ represents the spectral radius of a matrix [6]. The extension of this result to SLS's is stated in the following theorem.

Theorem 1 Consider the SLS (6), (7) characterized by a matrix E , a set of real matrices $\{A\}$ and a mapping Γ. 0 is an asymptotically stable equilibrium point of the SLS if and only if $\rho(\Phi(T,0)) < 1$ for all realizable periodic systems.

The conditions stated in this theorem may be difficult to check since the number of realizable periodic systems may be large. In the case that the characteristic polynomial of E is primitive, there is only one realizable periodic system and the conditions of the above theorem are easy to check [5].

An alternate approach to the stability analysis of SLS's, based on Liapunov theory, provides conditions on a matrix set $\{A\}$ which guarantee that *any* SLS with this parameter set is internally asymptotically stable.

Theorem 2 Suppose there exists a common Liapunov function for the set of systems

$$x_{k+1} = A_i x_k, \quad A_i \in \{A\} \tag{10}$$

Then *any* SLS with this parameter set is internally asymptotically stable.

This theorem states that, if a SLS has a parameter set $\{A\}$, and there exists a common Liapunov function for the set of systems (10), then we can conclude that 0 is an asymptotically stable equilibrium point of the SLS without considering the structure of the discrete-state system. This result follows directly from Liapunov's asymptotic stability theorem given the fact that a common Liapunov function for (10) is, by definition, time-invariant. For a discussion of common Liapunov functions see [9].

Sufficient conditions for the existence of common *quadratic* Liapunov functions for certain sets of systems are given in Theorem 3.

Theorem 3 Consider the set of systems (10) where each A_i is of the form

$$A_i = A_0 - BK_iC', \quad K_i \in \{K\} \tag{11}$$

where $A_0 \in R^{n \times n}$, $\rho(A_0) < 1$, $B, C \in R^{n \times p}$ and $\{K\}$ is a finite set of real $p \times p$ matrices. Suppose there exists a positive definite, symmetric matrix Q such that

$$CK_iC' > CK_i'GK_iC', \quad \text{for } \textit{all } K_i \in \{K\} \tag{12}$$

where

$$G := B'PB + (A'PB + \tfrac{1}{2}C)'Q^{-1}(A'PB + \tfrac{1}{2}C)$$

and where P is the unique positive definite solution to

$$A_0'PA_0 - P = -Q \tag{13}$$

Then $V(x_k) = x_k'Px_k$ is a common quadratic Liapunov function for the set of systems (10),(11).

This theorem can be simplified in the case where $B, C \in R^{n \times 1}$. In this case the application of the theorem becomes easier from a computational point of view.

Corollary Consider the set of systems (11) where $b := B \in R^{n \times 1}$, $c := C \in R^{n \times 1}$ and $\{K\}$ is a finite set of nonnegative real numbers. Let $\bar{K} := \max\{K_i\}$. If there exists a positive definite symmetric matrix Q such that

$$G > \bar{K} \tag{14}$$

Then $V(x_k) = x_k'Px_k$ is a common Liapunov function for the set of systems (10),(11).

These results were motivated by work done by Pearson on the discrete-time Lur'e problem [8] as well as work done by Anderson on the stability of control systems with multiple nonlinearities [1].

Observer Design for SLS's

In addressing the observer design problem for SLS's we shall assume full information about the system other than the state itself. That is, we will assume that the parameter matrix sets of the continuous-state system, the parameters of the discrete-state system and the mapping Γ. are all known. In addition we will assume that u_k, y_k and l_k are all available for measurement.

A full order, closed loop observer which provides estimates of the discrete and continuous states of a SLS is of the following form:

$$\hat{x}_{k+1} = A(\hat{j}_k)\hat{x}_k + B(\hat{j}_k)u_k + F(\hat{j}_k)(y_k - \hat{y}_k) \tag{15}$$

$$\hat{y}_k = C(\hat{j}_k)\hat{x}_k + D(\hat{j}_k)u_k \tag{16}$$

$$\hat{j}_{k+1} = E\hat{j}_k + H(l_k - \hat{l}_k) \tag{17}$$

$$\hat{l}_k = G\hat{j}_k \tag{18}$$

Here $\hat{x}_k, \hat{j}_k, \hat{y}_k, \hat{l}_k$ represent the states and outputs of the SLS observer, H is a gain matrix which must be determined and $F(\cdot)$ is a matrix-valued function which must also be determined to ensure that the observed continuous-state asymptotically approximates the continuous-state of the system.

The SLS observer consists of two subsystem observers, one for the discrete-state and one for the continuous-state. The error between the actual discrete-state and the discrete-state of the observer is given by

$$(j_{k+1} - \hat{j}_{k+1}) = (E - HG)(j_k - \hat{j}_k) \tag{19}$$

Since the discrete-state estimate is used to update the parameters in the continuous-state observer, the error in the discrete state must eventually go to zero. Otherwise, there would always be parameter errors in the continuous observer. In order to ensure that this happens we must choose H to make the matrix $(E-HG)$ nilpotent. Observability of the pair (G,E) is a sufficient condition for the existence of such an H. This follows from the general eigenvalue placement result of Mitter and Foulkes [7]. That is, observability of the pair (G,E) implies that there exists an H which places all the eigenvalues of $(E-HG)$ at zero.

If we assume that the error in the observed discrete-state goes to zero, the error in the observed continuous-state is described by the following SLS after some initial transient period:

$$(x_{k+1} - \hat{x}_{k+1}) = (A(j_k) - F(j_k)C(j_k))(x_k - \hat{x}_k) \tag{20}$$

$$j_{k+1} = Ej_k \tag{21}$$

In order to ensure asymptotic estimation of the continuous-state, we must choose a set $\{F\}$ such that, under the mapping Γ , 0 is an asymptotically stable equilibrium point of (20), (21).

Kono [4] has shown that, for T periodic, discrete-time, linear systems, complete reachability of the system is equivalent to the existence of a T periodic, memoryless state feedback which arbitrarily places the eigenvalues of the state transition matrix of the closed loop system over one period. By the duality of complete reachability and complete observablity we can extend this result so that it is useful in the observer design problem for SLS's.

Theorem 4 Suppose each realizable periodic system of a SLS is completely observable. Then there exists a set of real matrices $\{F\}$ such that, under the mapping Γ, 0 is an asymptotically stable equilibrium point of (20), (21).

This theorem follows from the fact that we can choose $\{F\}$ to ensure that each realizable periodic system of (20), (21) is stable.

Examples

Consider the SLS

$$x_{k+1} = A(j_k)x_k \tag{22}$$

$$j_{k+1} = e \cdot j_k \tag{23}$$

where the scalars e, $j_k \in GF(7)$ and $A(j_k) \in \{A\}$, a set of real 2×2 matrices.

(i) Supppose $e = 2$ and $\{A\} = \{A_1, A_2, A_3, A_4\}$, where

$$A_1 = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}, A_2 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, A_3 = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}, A_4 = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/3 \end{bmatrix}$$

The mapping $\Gamma : GF(7) \rightarrow \{A\}$ is given explicitly below.

$$0 \overset{\Gamma}{\rightarrow} A_4 \quad 1,3 \overset{\Gamma}{\rightarrow} A_1 \quad 2,6 \overset{\Gamma}{\rightarrow} A_2 \quad 4,5 \overset{\Gamma}{\rightarrow} A_3$$

It is not difficult to show that there are three realizable periodic systems for this particular SLS corresponding to the discrete-state sequences below.

$\{0, 0, 0, 0 \ldots\}$

$\{1, 2, 4, 1, 2, 4 \ldots\}$

$\{3, 6, 5, 3, 6, 5 \ldots\}$

Now, since $\rho(A(0)) < 1$, $\rho(A(1) \cdot A(2) \cdot A(4)) < 1$ and $\rho(A(3) \cdot A(6) \cdot A(5)) < 1$ we can conclude, from Theorem 1, that the SLS is stable.

(ii) We will consider the same SLS as above but change the mapping Γ to

$$0 \xrightarrow{\Gamma} A_4 \quad 1,3 \xrightarrow{\Gamma} A_1 \quad 2,6 \xrightarrow{\Gamma} A_3 \quad 4,5 \xrightarrow{\Gamma} A_2$$

In this case $\rho(A(1) \cdot A(2) \cdot A(4)) > 1$ and $\rho(A(3) \cdot A(6) \cdot A(5)) > 1$, hence any nonzero initial discrete- state leads to unbounded growth in the continuous state for nonzero x_0. That is, the SLS is unstable.

(iii) Suppose that a SLS of the form (22), (23) has a parameter set defined as follows:

$$\{A\} = \{A_1, A_2, A_3\}$$

$$A_i := A_0 - K_i bc' \quad , i = 1,2,3$$

with $K_1 = 1/3$, $K_2 = 1/4$, $K_3 = 1/8$ and

$$A_0 = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/4 \end{bmatrix} , b = \begin{bmatrix} 1 \\ 1 \end{bmatrix} , c = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

If we choose $Q = I$ the conditions of the corollary to Theorem 3 are met (i.e. $G > 1/3$) and we can conclude that a common quadratic Liapunov function exists for the set of systems

$$x_{k+1} = A_i x_k , \quad i = 1,2,3$$

From Theorem 2 it follows that *any* SLS with this parameter matrix set is stable.

Concluding Remarks

Due to the fact that the discrete-state of a SLS evolves autonomously, any SLS can be described and analyzed in terms of a family of periodically varying, continuous-state systems. If the SLS model is generalized to include coupling between the continuous and discrete state, it appears that analysis becomes much more difficult. For example, it is unlikely that the definition of stability presented in this paper would be adequate.

References

[1] Anderson, B.D.O., *Stability of Control Systems With Multiple Nonlinearities* , Journal of the Franklin Institute, Vol 282, No. 3, Sept. 1966, pp 155-160.

[2] Booth, T.L., *Sequential Machines and Automata Theory* , Wiley, New York, 1967.

[3] Gill, A., *Linear Sequential Circuits* , McGraw-Hill, New York, 1966.

[4] Kono, M., *Eigenvalue Placement in Linear, Periodic, Discrete-Time Systems* ,Int. J. Control, Vol 32, No. 1, Jan 1980, pp 149-158.

[5] McIntyre, C.M., *Switched Linear Systems: Stability and Observer Design* , M.A.Sc. Thesis, Dept. of Electrical Engineering, University of Waterloo, Waterloo, Ontario, Aug. 1983.

[6] Meyer, R.A. and C.S. Burrus, *A Unified Analysis of Multirate and Periodically Time-Varying Digital Filters* , IEEE Trans. Circuits and Systems, Vol CAS-22, No. 3, Mar. 1975, pp 162-168.

[7] Mitter S.K. and R. Foulkes, *Controllability and Pole Assignment for Discrete-Time Linear Systems Over Arbitrary Fields* , SIAM J. Control, Vol 9, No. 1, Feb. 1971, pp 1-7.

[8] Pearson, J.B., *A Note on the Existence of Liapunov Functions for a Class of Discrete-Time Systems* , IEEE Trans. Automatic Control, Vol AC-8, July 1963, pp 89-92.

[9] Vidyasagar, M., *Nonlinear Systems Analysis* , Prentice-Hall, Englewood Cliffs, 1978.

ON FEEDBACK EQUIVALENCE OF RETARDED AND NEUTRAL DELAY-DIFFERENTIAL EQUATIONS

MARK W. SPONG

School of Electrical Engineering, Cornell University
Ithaca, NY, 14853

ABSTRACT

In this paper we consider derivative feedback for the control of a class of neutral differential-difference equations. Specifically, given a neutral system with finitely many commensurable or non-commensurable point delays, we give conditions under which a transformation consisting of derivative feedback plus a linear change of coordinates exists such that the resulting system is a retarded delay system. We also show that the above problem is mathematically related to the deadbeat control problem over rings and also to the problem of simultaneous stabilization of rational transfer functions. The motivation for this work comes from some recent results on feedback stabilization of neutral systems[3],[11],[14], where it is shown that, in general, neutral systems cannot be stabilized without derivative feedback, that is, using only state feedback, unless the D-operator in the hereditary representation of the system is stable, and even if the D-operator is stable, one has only limited control over the margin of stability of the closed loop system.

I. INTRODUCTION

In this paper we consider the application of derivative feedback for the control of a class of neutral delay-differential equations with finitely many point delays (commensurable or non-commensurable). Derivative feedback has been considered previously in the context of the pointwise degeneracy problem for neutral systems[1]. The primary motivation for the present work comes form some recent work on feedback stabilization [3],[11],[14], where it is shown that, in general, neutral systems cannot be stabilized using state feedback alone unless the difference operator in the hereditary representation of the system is stable, and even then one has only limited control over the margin of stability of the resulting closed loop system.

EXAMPLE

Consider the following example of O'Connor and Tarn [11]:

$$\dot{x}(t) = \dot{x}(t-1) - x(t) + x(t-1) + u(t)$$

This system has eigenvalues at $\{-1, 2k\pi i, k \in Z\}$ and it is shown in [11] that no state feedback operator $K(x_t)$ exists such that

$$\dot{x}(t) = \dot{x}(t-1) - x(t) + x(t-1) + K(x_t)$$

is stable. However, if we set $u(t) = -\dot{x}(t-1) - x(t-1)$ then the closed loop system is

---

This research was partially supported by the National Science Foundation under grant ECS-8214262

$$\dot{x}(t) = -x(t).$$

This example vividly illustrates the desirability (and necessity) of implementing derivative feedback for the control of neutral systems. Retarded delay systems, by contrast (under the assumption of spectral controllability) may be stabilized with arbitrary stability margin using only state feedback [13].

Motivated by the above discussion we seek to characterize the class of neutral systems which are feedback equivalent under derivative feedback (in a sense to be made precise) to retarded delay-differential systems.

Using the theory of systems over rings we give conditions under which a transformation consisting of derivative feedback plus a linear change of coordinates exists such that the resulting system is a retarded delay system. We also show that the above problem is mathematically related to the deadbeat control problem over rings and also to the problem of simultaneous stabilization of rational transfer functions.

II. RING MODELS AND FEEDBACK TRANSFORMATIONS

The class of systems that we consider may be represented as

$$\sum_{i=0}^{N} D_i \dot{x}(t-\tau_i) = \sum_{i=0}^{N} A_i x(t-\tau_i) + \sum_{i=0}^{M} B_i u(t-\tau_i) \tag{1}$$

where $D_i, A_i \in \mathbb{R}^n$, $B_i \in \mathbb{R}^m$.

Introducing a delay operator z_i for each rationally independent generating the set $\{\tau_i\}$ we may write equation (1) as a so-called generalized linear system over the polynomial ring $R = \mathbb{R}[z]$,

$$D(z)\dot{x}(t) = A(z)x(t) + B(z)u(t) \tag{2}$$

where we are using the multi-index notation $z = (z_1, \dots, z_k)$. Equation (2) is an equation of neutral type if and only if $D(0)$ is non-singular. (2) is a retarded system if and only if $D = I$, and is said to be *neutral with formally stable D-operator* if and only if $\det D(z) \neq 0$ for all $z \in X \subset C^k$, where X is the unit polydisk in C^k. If $D(z)$ is formally then it is stable in the sense of Cruz and Hale [5], i.e.,

$$\det D(e^{-sh_1}, \dots, e^{-sh_k}) \neq 0 \tag{3}$$

for Re $s \geq 0$.

Given the neutral system (2) the following facts are well-known or can be obtained from references [3],[5],[6],[11],[12],[14].

1) If $D(z)$ is formally stable then the spectrum of the neutral system, i.e., the set

$$\{\lambda \in C \mid \det(\lambda D(e^{-sh_1}, \dots, e^{-sh_k},) - A(e^{-sh_1}, \dots, e^{-sh_k})$$

consists only of point spectrum,(eigenvalues),and contains only finitely many points in the right half of the complex plane Re $s \geq 0$. Hence modal techniques, such as the result of Pandolfi [12] can then be used for stabilization.

2) If D(z) is not stable then no state feedback law alone exists which can stabilize the system, since the spectrum of the D(z) operator is invariant under state feedback.

3) Even if D(z) is stable and the system is spectrally controllable there is limited control over the margin of stability of the closed loop system for any state feedback law.

4) If D(z) is unimodular, i.e. det $D(z) \equiv \text{constant} \neq 0$ for all $z \in C^k$ then (2) is called pseudo-neutral and Kappel [8] has shown that by a suitable change of basis that (1) is equivalent to the retarded delay system

$$\dot{w}(t) = A(z)D^{-1}(z)w(t) + B(z)u(t) \qquad (4)$$

in the sense that solutions of (1) and (4) coincide for $t > T$, for some T.

Based on the above discussion we state the following:

Problem 1:

Find L(z) over R[z] of suitable dimensions such that

$$\tilde{D}(z) := D(z) + B(z)L(z) \qquad (5)$$

is unimodular.

For, if such an L(z) exists, a feedback control law of the form

$$u(t) = -L(z)\dot{x}(t) + v(t) \qquad (6)$$

when substituted into (2) leads to the pseudo-neutral system

$$\tilde{D}\dot{x}(z) = A(z)x(t) + B(z)v(t) \qquad (7)$$

Thus in the next section we shall derive various conditions for the existence of an L(z) which solves Problem 1. We see that Problem 1 is mathematically related to the problem of simultaneous stabilization of rational transfer functions [15] and shall see a further connection to the deadbeat control problem over the ring R.

III. MAIN RESULTS

NONCOMMENSURABLE DELAYS

From the more or less obvious fact that for any L(z)

$$\text{rank}[D(z) + B(z)L(z) \ , B(z)] = \text{rank}[D(z) \ , B(z)]$$

it follows that a necessary condition for the existence of a solution to problem 1 is that

$$\text{rank}[D(z) \ , B(z)] = n \quad \text{for all } z \in C^k$$

which is of course the statement that D(z), B(z) must be left coprime. We shall always assume, then, that D, B in (2) are left coprime. That this condition is not sufficient for the existence of a solution to Problem 1, however, is easily demonstrated by the following scalar example

$$[D(z)\ ,\ B(z)] = [1 + z_1 \mid z_2]$$

since there is no polynomial q such that $z_1 + qz_2$ is constant. If rank B(z) is full for all $z \in C^k$, however, then the condition is sufficient. We state this as

Theorem 1:

Suppose rank B(z) = m for all z. Then Problem 1 has a solution if and only if rank [D(z),B(z)] = n for all $z \in C^k$.

Proof: Since B(z) has rank m identically in z we may assume that a change of basis in the input space has been performed to put B(z) into the form

$$B(z) = \begin{bmatrix} I_{m\times m} \\ 0 \end{bmatrix} .$$

Then

$$D + BL = \begin{bmatrix} D_1 + L_1 & D_2 + L_2 \\ D_3 & D_4 \end{bmatrix} \tag{8}$$

where

$$[D(z)\ ,\ B(z)] = \begin{bmatrix} D_1 & D_2 & I \\ D_3 & D_4 & 0 \end{bmatrix}$$

Problem 1 then has a solution if and only if $[D_3\ ,\ D_4]$ is unimodular, since in (8) a unimodular completion of $[D_3\ ,\ D_4]$ can always be found over R. QED

Now, suppose we write D(z) = I - P(z), with P(0) = 0 (since D(z) must be atomic at zero). Then over the ring of formal power series,

$$D^{-1}(z) = \sum_{i=0}^{\infty} P^i(z).$$

Therefore if P(z) is nilpotent $D^{-1}(z)$ is a polynomial in P, hence polynomial in z, which is equivalent to D(z) being unimodular. Thus related to problem 1. i.e., a sufficient condition for its solution, is the problem of finding L(z) so that P(z) + B(z)L(z) is nilpotent. This is then just the deadbeat control problem over R[z], and the problem becomes one of assigning λ^n as the characteristic polynomial of P + BL by suitable choice of L.

The problem of pole assignment over R[z] is known to be a difficult problem. Unlike the case of systems over fields, for example, ring reachability of the pair (P(z), B(z)) is not sufficient for arbitrary pole assignment. For a solution to Problem 1, however, we require only that all the closed loop poles be assigned the value zero. Necessary and sufficient conditions for this are unknown at present. We shall give here some sufficient conditions. For example it follows from the work of Byrnes[2],[3] that if the pointwise Kronecker indices of a reachable pair (P(z),B(z)), are constant in z then the pair is in fact coefficient assignable over R and hence problem 1 has a solution in this case.

More specifically, let us define submodules $M_1, \ldots, M_k$ of R^n as follows:

$$M_j = \mathrm{span}[B, PB, P^2B, \ldots, P^{j-1}B], \; j=1,\ldots,n$$

Then $M_1 \subseteq M_2 \subseteq \cdots \subseteq R^n$, and one can show that problem 1 has a solution if each M_i is a direct summand of M_{i+1}. For it follows, in this case that $(P(z),B(z))$ may be put into Brunovsky canonical form via feedback and coordinate transformation over R, and P in Brunovsky form is nilpotent.

COMMENSURABLE DELAYS

In the case of commensurate delays , i.e. $k=1$, R is a p.i.d. and we may give both necessary and sufficient conditions for the solution to Problem 1.

Let us denote by $\gamma(f)$ the degree of a polynomial $f \in R$. Then, since D and B are left coprime, we can say that $L(z)$ is a solution to Problem 1 if and only if

$$\gamma(\det(D+BL)) = 0. \tag{9}$$

Now, given polynomials d,b there exist unique polynomials l,r such that $d = bl + r$ with $\gamma(r) < \gamma(b)$. We may now state our main result as

Theorem 2: Let $d = \det D(z) \neq 0$ and let b be the greatest common divisor of the minors of $B(z)$. Assume $D(z)$, $B(z)$ are left coprime. Then a solution to Problem 1 exists if and only $\gamma(d \bmod (b)) = 0$.

Proof: Under the above assumptions, this is a direct application of the result, proved in [15] (Lemma 4.2), and valid over any proper Euclidean domain which states that with the above notation

$$\min_{L \in R^{m \times n}} \gamma(\det(D + BL)) = \min_{l \in R} \gamma(d + bl) \quad \text{QED.}$$

EXAMPLE

$$\text{Set} \quad D(z) = \begin{bmatrix} 1 & -z \\ z & 1 \end{bmatrix}, \quad B(z) = \begin{bmatrix} z^k \\ 0 \end{bmatrix}$$

Then $d = 1 + z^2$, $b = z^k$. Since

$$1+z^2 = z^k z^{2-k} + 1$$

for $k \leq 2$, we see that $\gamma(d \bmod (b)) = 1$ if and only if $k \leq 2$. Notice that in this case

$$P = \begin{bmatrix} 0 & z \\ -z & 0 \end{bmatrix}$$

and it is easily shown that there is an L such that $P + BL$ is nilpotent if and only if $k \leq 1$.

In the commensurable delay case, since $P(0) = 0$, we may write $P(z) = zS(z)$ and consider the difference equation over $R[z]$

$$D(z)y(t) = zB(z)u(t) \tag{10}$$

which we write as

$$y(t) = zS(z)y(t) + zB(z)u(t) \qquad (11)$$

Setting $\sigma = z^{-1}$ = forward shift operator and multiplying (11) by σ we obtain

$$y(t+1) = S(z)y(t) + B(z)u(t) \qquad (12)$$

It thus follows from Morse's Theorem [10] that if the pair $(S(z),B(z))$ is R-reachable there exists $L_1(z)$ over R such that

$$\det[\sigma I-S(z)+B(z)L_1(z)] = \sigma^n \qquad (13)$$

Setting $L(z) = zL_1(z)$ we have

$$\det D = \det[I-zS(z)+zB(z)L_1(z)] = z^n \det[\sigma I-S+BL_1] = 1. \qquad (14)$$

SOME SPECIAL CASES

(i) Consider the following special case of (1)

$$\dot{x}(t)-A_{-1}\dot{x}(t-h) = A_0x(t)+A_1(t-h)+Bu(t) \qquad (15)$$

Then $(S(z),B(z)) = (A_{-1},B)$ and it is known [7] that reachability of (A_{-1},B) is implied by function space controllability in the Sobolev space W_{12} of (15). ∴Under the assumption that (15) is exactly controllable in W_{12} we have shown that there is a derivative feedback law under which the closed loop system is pseudo-neutral.

(ii) Consider the scalar n-th order neutral differential-difference equation

$$\frac{d^n y}{dt^n} + \sum_{j=0}^{n} \sum_{i=1}^{T} \alpha_i^j \frac{d^j y(t-\tau_i)}{dt^j} = u(t)$$

which we may write over R[z] as

$$(1+p_n(z))\frac{d^n y}{dt^n} + \sum_{j=0}^{n-1} p_j(z)\frac{d^j y}{dt^j} = u(t)$$

which, in turn, may be written as the first order system

$$\dot{x}(t) = \begin{bmatrix} 0 & \cdot & \cdot & 0 \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ 0 & \cdot & \cdot & -p_n(z) \end{bmatrix} \dot{x}(t) + \begin{bmatrix} 0 & 1 & 0 & \cdots & \\ 0 & 0 & 1 & \cdots & \\ & & & & \\ -p_0(z), & \cdot & \cdot & \cdot & -p_{n-1}(z) \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \cdot \\ \cdot \\ \cdot \\ 1 \end{bmatrix} u(t).$$

Thus $[D(z),B(z)] =$

$$\left[\begin{array}{c|c} I_{(n-1)x(n-1)} & 0 \\ & \\ * & 1 \end{array}\right]$$

and by Theorem 2 Problem has a solution. In fact by inspection we see that with

$$u(t) = \left[0,\ldots,-p_n(z)\right]\dot{x}(t) + v(t)$$

we obtain the retarded system system

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 & & 0 \\ 0 & & 0 & \\ & & & 1 \\ -p_0 & & \cdots & -p_{n-1} \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \cdot \\ \cdot \\ \cdot \\ 1 \end{bmatrix} v(t).$$

D-STABILIZATION

For many if not most applications it may be sufficient simply to stabilize the D-operator appearing in the neutral system, i.e., choose $L(z)$ such that $\det(D(z) + B(z)L(z)) \neq 0$ for $z \in X$ = unit polydisk in C^k.

Set $S = \{ p \mid p \neq 0 \text{ in } X \}$ and let $R_S = S^{-1}R$ denote the localization,

$$R_S = \{q/p \mid q \in R, p \in S\}.$$

Then we state

Problem 2:

Find $L(z) \in R_S^{mxn}$ such that $D(z) = D(z) + B(z)L(z)$ is unimodular over R_S, i.e., $D(z)$ is formally stable.

With this set-up all of our previous results remain valid simply by replacing R by R_S and C^k by X. The proof of this statement, however, is non-trivial. For example, the extension of Theorem 1 requires that R_S be a projective-free ring, the proof of which is difficult. See [3] for a complete proof.

In order to have a useful result for neutral systems we must show that any solution to problem 2 has a realizable delay system interpretation. We may do this as follows:

Consider the neutral system (2) and consider the transformations

$$x(t) = P(z)w(t) \tag{16}$$

$$u(t) = -Q(z)\dot{w}(t) + v(t) = -QP^{-1}\dot{x}(t) + v(t) \tag{17}$$

$$= -L(z)\dot{x}(t) + v(t)$$

where $\det P(z) \in S$, i.e., $P(z)$ is R_S unimodular. In this case the results of Kappel [8] are no longer valid and solutions of the two neutral systems

$$D(z)\dot{x}(t) = A(z)x(t) \tag{18}$$

$$D(z)P(z)\dot{w}(t) = A(z)P(z)w(t) \tag{19}$$

will not, in general, agree at any finite time t. However, since $P(z)$ is formally stable, it follows from [6], see also [3] that for any initial states,

$$x_t \rightarrow 0 \text{ as } t \rightarrow \infty$$

if and only if

$$w_t \rightarrow 0 \text{ as } t \rightarrow \infty,$$

i.e., stability is preserved under the change of coordinates (16). Thus we may interpret any solution of Problem 2 as consisting of an R_S -unimodular change of coordinates followed by feedback. If the system with state vector w is stabilized it then follows form (16) since P(z) is stable that the system with state x is stable for any initial condition.

IV. REFERENCES

[1] Asner, and Halanay, "Delay-Feedback using Derivatives for Minimal Time Linear Control Problems", J. Math. Anal. Appl. - v. 48, (1974), 257-263.

[2] Byrnes, "On the Control of Certain Infinite-Dimensional Linear Systems by Algebro-Geometric Techniques," Amer. J. Math., 100, 1333-1381 (1978).

[3] Byrnes, Spong, and Tarn, "A Several Complex Variables Approach to Feedback Stabilization of Neutral Delay-Differential Equations", to appear in Math. Sys. Theory.

[4] Bumby, Sontag, Sussmann, Vasconcelos, "Remarks on the Pole-Shifting Problem over Rings", J. Pure and Appl. Algebra, 20, 113-127 (1981).

[5] Cruz, and Hale, "Stability of Functional Differential Equations of Neutral Type", J. of Diff. Equations, 7, 334-355 (1970).

[6] Hale, *Functional Differential Equations*, Springer-Verlag, New York, 1972.

[7] Jacobs, and Langenhop, "Criteria for Function Space Controllability of Linear Neutral Systems", SIAM J. Cont. and Opt., 14, 1009-1048 (1976).

[8] Kappel, "Degenerate Difference-Differential Equations: Algebraic Theory", J. Diff. Eq., 243, (1977), pp 99-126.

[9] Moore and Steger, "Some Results on Completability in Commutative Rings", Pacific Journal of Math. , vol. 37, no. 2 , (1971).

[10] Morse, "Ring Models for Delay-Differential Systems", Automatica, 12, (1976), 529-531.

[11] O'Connor and Tarn. "On Stabilization by State Feedback for Neutral Differential Difference Equations", IEEE Trans. Aut. Cont., 1983.

[12] Pandolfi. L., "Stabilization of Neutral Functional Differential Equations," J. Opt. Theory and Appl., 20 (1976).

[13] Pandolfi. L., "On Feedback Stabilization of Functional Differential Equations," Bolletino U.M.I. vol. 4 no. 11, pp. 626-635, (1975).

[14] Salamon, "On Control and Observation of Neutral Systems", Doctoral Dissertation, Bremen, 1982.

[15] Vidyasagar and Wiswanadham, "Algebraic Design Techniques for Reliable Stabilization", IEEE Trans. Aut. Cont. (Oct. 1982), 1085-1096.

An Experimental Study of Bin Packing*

Jon Louis Bentley
David S. Johnson
Bell Laboratories
Murray Hill, NJ 07974

Tom Leighton

Department of Mathematics and Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139

Catherine Cole McGeoch

Department of Computer Science
Carnegie-Mellon University
Pittsburgh, PA 15213

Abstract

Most previous research on bin packing has dealt with the worst-case performance of various heuristics. In this paper we describe an experimental study of the average performance of the First Fit and First Fit Decreasing heuristics when the inputs are uniformly distributed over various ranges. Our data leads naturally to several conjectures, all of which imply that both heuristics perform much better than previously suspected.

1. Introduction

The bin packing problem is to place a given set of weights into a minimal number of bins under the constraint that the sum of the weights in any bin cannot exceed unity. The problem and several generalizations arise naturally in a host of applications. Because of the great volume of previous work on the subject, we will cite as a reference only Coffman, Garey and Johnson [1983]; readers interested in more detail may consult the over one hundred references cited therein.

Since bin packing is NP-complete, previous work has concentrated on finding approximation algorithms guaranteed to produce packings quantifiably close to optimal. In one sense, this paper describes a continuation of that line: we investigate two classical approximation algorithms for the problem. Our work differs from most previous papers in two important ways. First, we are concerned with the expected performance of the algorithms, rather than their performance in the worst case. Second, our approach is experimental rather than analytic; more limited experiments have been described by Johnson [1973], Maruyama, Chang and Tang [1977] and Ong, Magazine and Wee [1982]. Several of the conjectures resulting from our experiments have been proved, however; see Bentley et al. [1983].

---

* Portions of this research were performed while Leighton and McGeoch were visiting Bell Laboratories. McGeoch was supported in part by the Office of Naval Research under Contract N00014-76-C-0370 and in part by a National Science Foundation Graduate Fellowship.

Sections 2 and 3 describe our experiments for two particular heuristics. Section 4 is a discussion of the experimental methodology.

2. First Fit Decreasing

The First Fit (FF) heuristic packs each weight by scanning a sequence of initially empty bins and placing the weight in the first bin in which there is room. The First Fit Decreasing (FFD) heuristic does the same but initially sorts the weights so that they are processed in nonincreasing order by weight. In this section we will study the performance of the FFD heuristic when the input weights are drawn uniformly on the range $[0,u]$, where the upper bound u is allowed to vary between 0 and 1.*

Previous work has typically measured the performance of a heuristic by the ratio of the number of bins used by the heuristic to the number of bins needed by an optimal packing. We will instead measure the total amount of "empty space" in the bins (equivalently, the number of bins used minus the sum of the weights). This quantity is easier to measure (the above ratio would require computing the optimal packing), and the empty space is an upper bound on the difference between bins used and bins needed. We will write $E_H(N,u)$ to denote the expected empty space when N weights uniform from $[0,u]$ are packed by heuristic H.

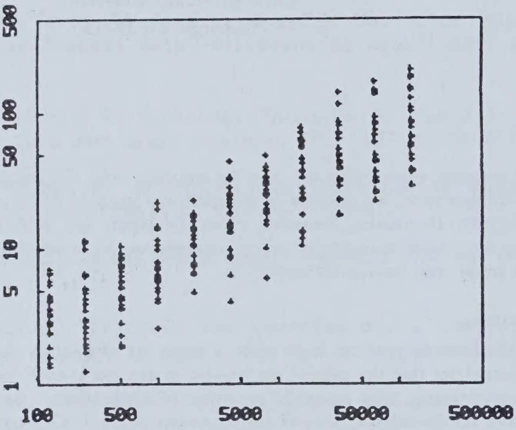


Figure 2.1.
FFD
$u=1$

Figure 2.1 displays the empty space in 275 FFD packings of weights uniform from $[0,1]$. The x coordinate of a point represents N, and the y coordinate represents the empty space observed (both on logarithmic scales). Twenty-five random packings were computed for each N from 125 doubling to 128,000. The graph contains two important messages. The first is that the FFD packings have very little empty space: even though the packing 128,000-weight packings used roughly 64,000 bins, the total amount of empty space was around 100. (Johnson's [1974] worst-case bound guarantees only that empty space is less than $2/9 \times 64000+4$, or a little over 14,200).

---

* We chose to investigate the uniform distribution because it is an accurate first-order model for several applications and it appeared to be the most straightforward to analyze (either experimentally or mathematically). We initially restricted the lower bound of the uniform distribution to zero for experimental ease; we were later able to analyze that case mathematically without recourse to experiment.

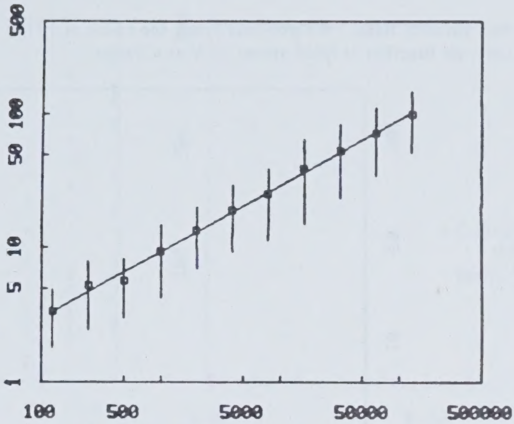


Figure 2.2.
FFD means
$u=1$

The second fact apparent in Figure 2.1 is that the amount of empty space grows smoothly as a function of N. This is more obvious in Figure 2.2, which summarizes Figure 2.1: for each value of N, the mean amount of empty space is denoted by a box and the standard deviation is shown by a vertical bar. The line near the means is the least-squares regression fit of the points; it has the form $0.303N^{.4959}$. This leads naturally to our first conjecture.

Conjecture: If N weights uniform from $[0,1]$ are packed by FFD, the expected amount of empty space is roughly $0.3N^{1/2}$; equivalently, $E_{FFD}(N,1) \sim 0.3N^{1/2}$.

This conjecture was first made by one of the authors (JLB) in 1980, based on earlier, more limited experiments performed jointly with Mark Faust. It was communicated to George Lueker in 1981, who proved it correct in Lueker [1982].

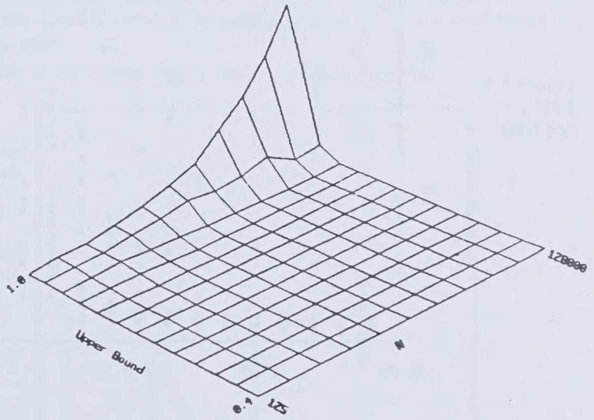


Figure 2.3
FFD means

The characterization of the single-variable function $E_{FFD}(N,1)$ still leaves open the behavior of the two-variable function $E_{FFD}(N,u)$. Our first attempt to understand that function was through the three-dimensional representation of Figure 2.3; it plots empty space (linear scale in the z-axis) as a function of upper bound u (linear scale) and N (logarithmic scale). Unfortunately, the figure is too coarse to reveal relevant details. We will therefore study various slices through the graph that leave one variable fixed. We previously saw the result of fixing u at unity and varying N; we will now study the function at fixed values of N as u varies.

Figure 2.4
FFD
$N=1000$

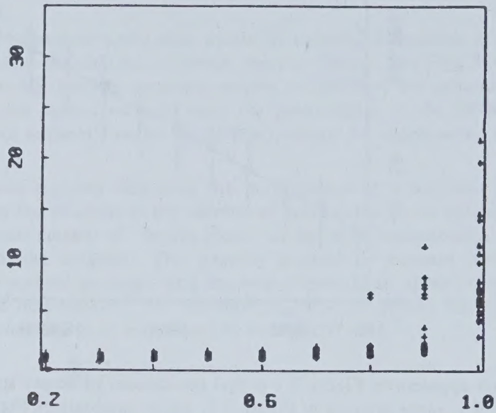
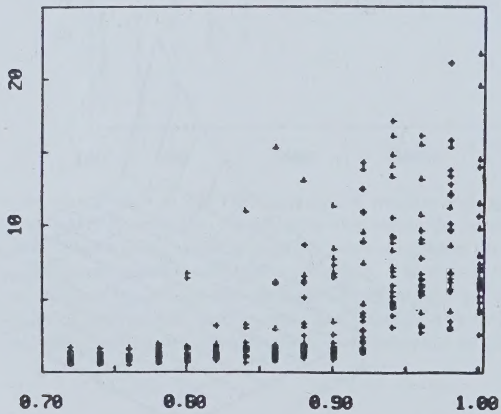


Figure 2.4 fixes N at 1000 and varies u at the multiples of 0.1 from 0.2 to 1.0; each value of u has twenty-five points that represent the empty space in random packings. (In all graphs in this paper, the y coordinate denotes empty space and the x coordinate denotes either N or u.) The graph shows that empty space is consistently quite small until u reaches 0.8, when the variance starts to increase; by the time u reaches 1, both the mean and the variance have risen dramatically.

Figure 2.5
FFD
$N=1000$



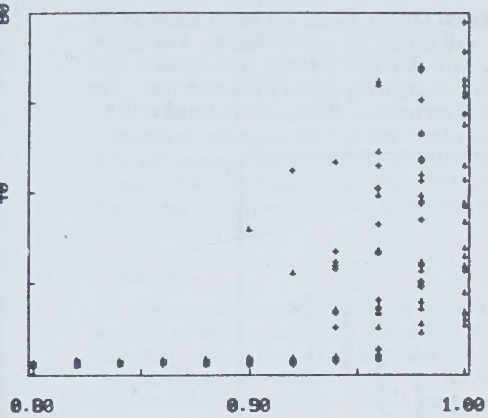


Figure 2.6
FFD
$N = 16{,}000$

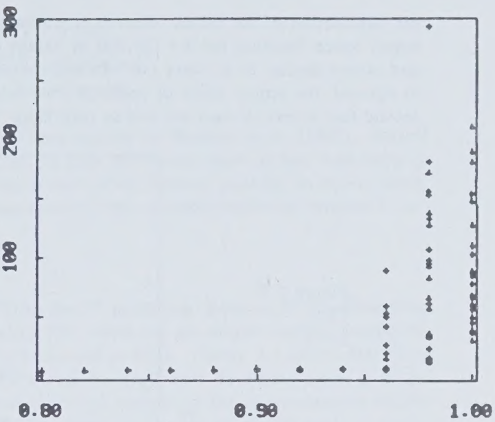


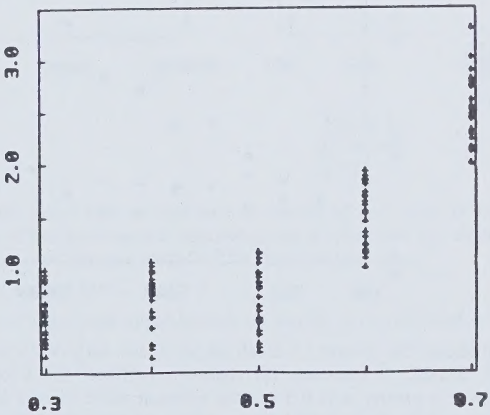
Figure 2.7
FFD
$N = 128{,}000$

Because the sampling at multiples of 0.1 may be too coarse, Figure 2.5 describes additional experiments at increments of u of 0.02 in the range [0.72,1]; that graph displays the same kind of behavior. Figures 2.6 and 2.7 show similar slices of the function at $N = 16000$ and $N = 128000$. All graphs have the same character: as long as u is less than a certain "critical point" the empty space is tightly clustered near zero, at the critical point the variance increases, and by the time u reaches unity the empty space is broadly dispersed around $0.3N^{1/2}$. This suggests the following.

> **Conjecture:** There is a critical region near the line $u = 1$ in the N,u-plane such that in that region the empty space quickly grows to be proportional to $N^{1/2}$ while outside the region the empty space is quite small.

The precise characterization of the critical region remains an open problem.

Figure 2.8
FFD
N=128,000



The details of empty space outside the critical region are hidden in the previous graphs by the domination of the values where *u* is within the critical region. Figure 2.8 shows a slice of the empty space function for $N = 128{,}000$ at values of *u* well outside the critical region. This graph, and others similar to it, carry two obvious messages. First, the FFD packings are incredibly close to optimal: the empty space of packings outside the critical region is typically less than four. The second fact is less obvious but just as surprising: there is apparently a discontinuity around $u = 0.5$.

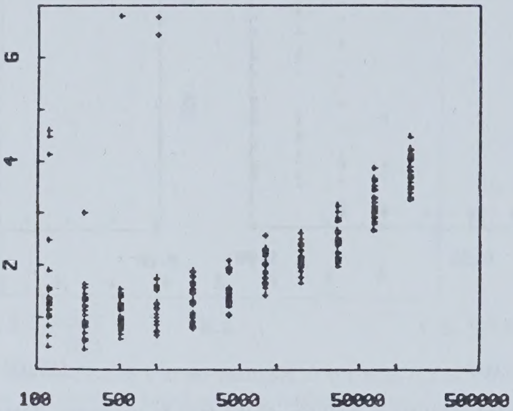


Figure 2.9
FFD
$u = .8$

Figure 2.9 fixes *u* at 0.8 and varies *N*. This graph shows the effect of the critical region: at values of *N* within the region (*N* less than 1000 for $u = 0.8$) there are occasional outliers, while above $N = 1000$ the empty space is clustered. The growth of the graph led us to conjecture originally that the empty space grows in this range as $\log N$, but the slight concavity upwards together with a deeper theoretical understanding led us to revise the conjecture (we will report that shortly).

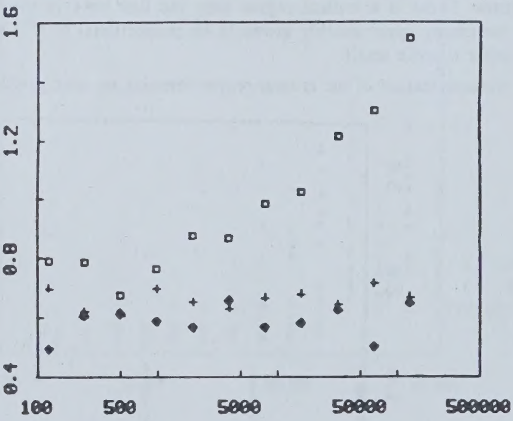


Figure 2.10
FFD means
$u = .4, .5, .6$

Figure 2.10 shows the growth of mean empty space with *N* for *u* fixed at 0.4 (dots), 0.5 (crosses) and 0.6 (boxes). It illustrates the slightly superlinear (on a log scale) growth of empty space for the value of *u* greater than 0.5 and the constant value when *u* is at most 0.5. Combining these two functions with Lueker's [1982] theorem for $u = 1$ gives the following.

Conjecture. The empty space function behaves as

$$E_{FFD}(N,u) = \begin{cases} \theta(N^{1/2}) & \text{if } u=1 \\ \theta(N^{1/3}) & \text{if } 1/2<u<1 \\ \theta(1) & \text{if } u \le 1/2 \end{cases}$$

The second and third parts of this conjecture have been proved by Bentley et al. [1983]. Figure 2.8 (and several other graphs like it) show that if $u<1/2$ then the wasted space is less than unity in roughly 75% of the packings; thus FFD produces a certifiably optimal packing in those cases. Tightening the constants implicit in the above thetas to reflect the observed behavior remains a formidable open problem.

3. First Fit

We will now turn our attention from the FFD to the FF heuristic. Because FF is online (that is, it places each weight into a bin as it is seen, while FFD must see all weights before placing the first one), we would expect that FF does not perform as well as FFD. Figure 3.1 shows that to be the case in the slice $E_{FF}(N,1)$. At $N=128{,}000$, FF's nine hundred units of empty space is larger than the one hundred units of FFD, but is still less than two percent of the approximately 64,000 bins used (and fifty times better than Johnson's [1974] worst-case bound of 45,000 units of empty space).

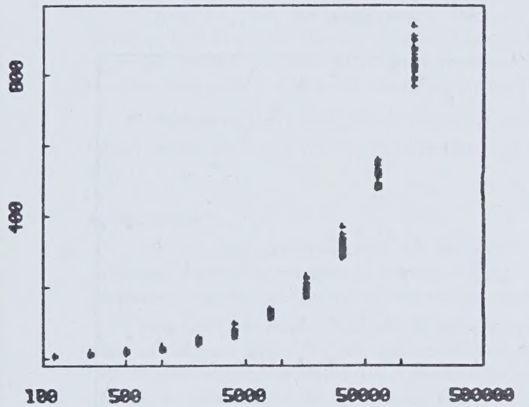


Figure 3.1
FF
$u=1$

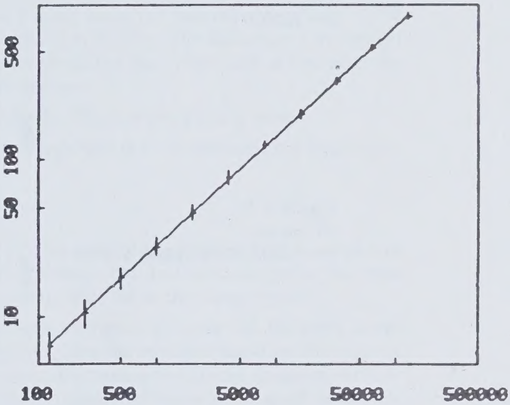


Figure 3.2
FF means
$u=1$

Figure 3.2 plots the means and the standard deviations of the points in Figure 3.1 on a log-log scale. The close fit of the least-squares regression line implies that the empty space is close to a power law; the parameters of the line ($0.224N^{.7012}$) yield the following.

Conjecture: $E_{FF}(N,1) \sim 0.22N^{.7}$.

Bentley et al. [1983] show lower and upper bounds on $E_{FF}(N,1)$ proportional to $N^{.5}$ and $N^{.8} \log N$, respectively.

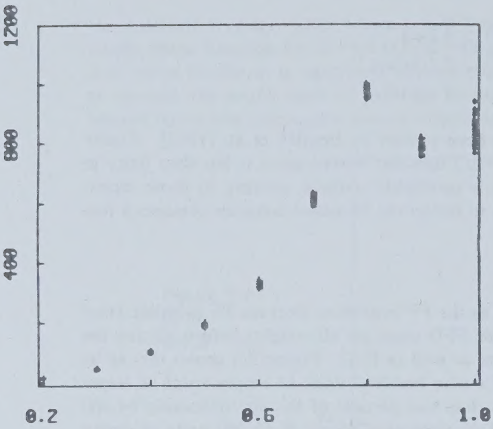


Figure 3.3
FF
$N = 128000$

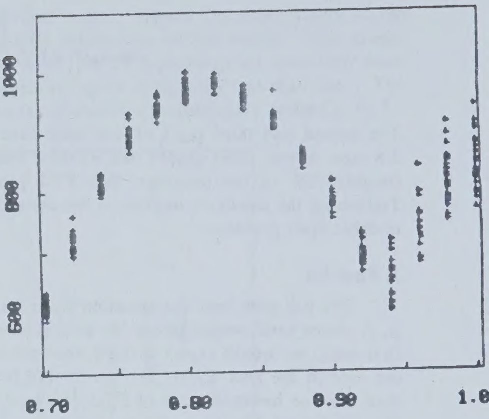


Figure 3.4
FF
$N = 128000$

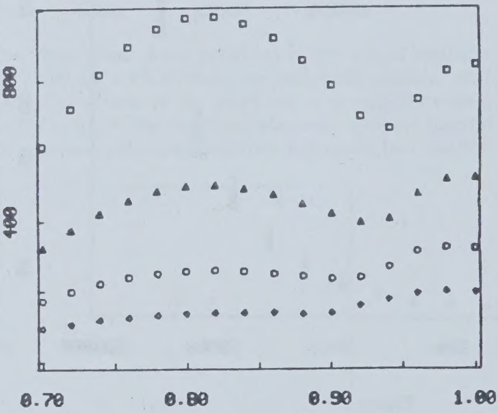


Figure 3.5
FF means
$N = 16000, 32000, 64000, 128000$

Figure 3.3 plots the slice $E_{FF}(128000,u)$ at values of u from 0.2 to 1.0. Because the growth of that function is smooth until the hiccup near 0.9, we ran additional experiments with denser values of u between 0.7 and 1.0; the resulting graph is in Figure 3.4. That graph displays a surprising nonmonotonicity of empty space as a function of u: there is less empty space when FF packs weights uniform from [0,0.94] than when it packs weights from either [0,0.8] or [0,1]. (Karmarkar [1982] has proved analytically that the Next-Fit heuristic has a similar nonmonotonicity.) Apart from the nonmonotonicity, this graph does have a similarity to the corresponding graph for FFD: the variance is tight at lower values and increases as u approaches 1. Figure 3.5 summarizes Figure 3.4 and three other slices at various values of N (the bottom function is for $N = 16000$, and the functions increase as N increases). That figure leads to the following.

Conjecture: The empty space function $E_{FF}(N,u)$ is nonmonotone in u, and the nonmonotonicity becomes more pronounced as N increases.

This conjecture is still open.

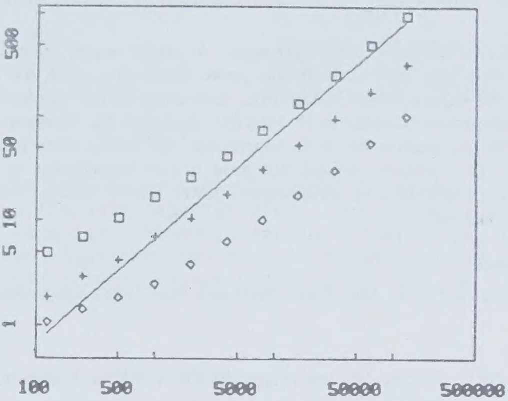


Figure 3.6
FF
$u = .4, .6, .8$

Figure 3.6 plots the mean empty space on a log-log scale for several values of u as N increases (the functions increase as u increases); the line is at $N/150$. The squares are the means when the weights are from $[0,.8]$; they appear to converge to the line. The case is less clear for the other two values of u; hence the following weak conjecture.

Conjecture: There exist some values of u such that $E_{FF}(N,u)$ grows linearly in N.

Larger values of N may have to be investigated if this conjecture is to be strengthened experimentally.

4. Discussion

The conclusions of this paper are obvious: the performance of the FFD and FF heuristics on uniformly distributed weights is extremely good. The future work is also clear: prove the open conjectures. In this section we will discuss the methodology that led to the conjectures.

Program Correctness. Counter-intuitive graphs such as Figure 3.5 raise the terrifying possibility of program bugs. We are quite confident, however, that the data presented in this paper is free of such irksome artifacts, for a number of reasons. We used principles of program verification as we constructed the programs, and tested the programs extensively on small examples. Where appropriate, we tested intermediate data for pseudo-randomness using several of the tests described by Knuth [1982]. Furthermore, we replicated the key experiments twice, varying the following in two implementations: computer type, operating system, programming language, word size, random number generator type, sorting algorithm, and algorithm for placing the weights in bins. The implementer of each program did not read the other program. Simple nonparametric tests confirmed the hypothesis that the output of both programs was from the same underlying distribution.

Program Efficiency. We were able to gather extensive data on large packings only because our primary simulation program was quite efficient. Our first approach to efficiency was algorithmic: we used a linear-time algorithm for generating sorted lists of random numbers similar to those of Bentley and Saxe [1980] and a logarithmic-time algorithm for placing weights in bins described by Johnson [1973,1974]. These algorithms gave a straightforward program, which we then sped up by a factor of two using techniques described by Bentley [1982]. The result was a program that could generate and pack by FFD 128,000 weights uniform from [0,1] in about sixty seconds. Altogether, the experiments described in this paper required a couple of days of VAX-11/750† CPU time.

---

† VAX is a Trademark of Digital Equipment Corporation.

Data Analysis. Both our exploration of the original data and the presentation in this paper have been greatly influenced by the books by Chambers, Cleveland, Kleiner and Tukey [1983] and by Tufte [1983].

The Interaction of Theory and Experiment. A simple model postulates that experimentalists generate conjectures and then theoreticians prove them; this paper has been presented in those terms. In fact, we experienced a much richer interaction in our research than is apparent in this paper and its companion (Bentley et al. [1983]). Although the theorems in the companion paper stem directly from conjectures based on experiment, the theory often had a substantial impact on the experiment. For instance, we did not have to run experiments in which the weights were drawn from uniform distributions with nonzero lower bounds because we are able to analyze that case with theoretical techniques.

Acknowledgements

The contributions of Al Aho, Mark Faust and Paul Tukey are gratefully acknowledged.

References

Bentley, J. L. [1982]. *Writing Efficient Programs*, Prentice-Hall, Englewood Cliffs, NJ.

Bentley, J. L., D. S. Johnson, F. T. Leighton, C. C. McGeoch, and L. A. McGeoch [1983]. "Some unexpected results about the expected behavior of bin packing algorithms", in preparation.

Bentley, J. L. and J. B. Saxe [1980]. "Generating sorted lists of random numbers", *ACM Transactions on Mathematical Software 6*, 3, September 1980, pp. 359-364.

Chambers, J. M., W. S. Cleveland, B. Kleiner and P. A. Tukey [1983]. *Graphical Methods in Data Analysis*, Wadsworth International Group, Belmont, CA.

Coffman, E. G., Jr., M. R. Garey and D. S. Johnson [1983]. "Approximation algorithms for bin-packing — an updated survey", to appear. (Copies are available from the authors; an earlier version by Garey and Johnson appears in *Analysis and Design of Algorithms in Combinatorial Optimization*, edited by B. Ausiello and M. Lucertini, Springer-Verlag, NY, 1981, pp. 147-172.)

Johnson, D. S. [1973]. Near-optimal bin packing algorithms. Ph.D. Thesis, Massachusetts Institute of Technology, June 1973. Project MAC TR-109.

Johnson, D. S. [1974]. "Fast algorithms for bin packing", *Journal of Computer and System Sciences 8*, pp. 272-314.

Karmarkar, N. [1982]. "Probabilistic analysis of some bin-packing problems", *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, pp. 107-111.

Knuth, D. E. [1982]. *The Art of Computer Programming, volume 2: Seminumerical Algorithms*, Addison-Wesley, Reading, MA.

Lueker, G. S. [1982]. "An average-case analysis of bin packing with uniformly distributed item sizes", University of California at Irvine Department of Information and Computer Science Technical Report #181, February 1982, 18 pp.

Maruyama, K., S. K. Chang and D. T. Tang [1977]. "A general packing algorithm for multidimensional resource requirements", *International Journal of Computer and Information Sciences 6*, 2, pp. 131-149.

Ong, H. L., M. J. Magazine and T. S. Wee [1983]. "Probabilistic analysis of bin packing heuristics", to appear in *Operations Research*.

Tufte, E. R. [1983]. *The Visual Display of Quantitative Information*, Graphics Press, Cheshire, CT.

VARIABLE SIZED BIN PACKING

D. K. FRIESEN
Computer Science Dept.
Texas A & M University
College Station, TX 77843

M. A. LANGSTON
Computer Science Dept.
Washington State University
Pullman, WA 99164-1210

ABSTRACT

In the classic bin packing problem one seeks to pack a list of pieces in the minimum space using unit capacity bins. This paper addresses the more general problem in which a fixed collection of bin sizes is allowed. Efficient approximation algorithms are described and analyzed which guarantee asymptotic worst-case performance bounds of 2, 3/2 and 4/3 respectively.

1. INTRODUCTION

Consider a finite collection of bin sizes and an inexhaustible supply of bins of each size. The problem we investigate is that of packing a list of pieces into bins so as to minimize the total space used in the packing. If the cost of a bin is proportional to its size then our objective corresponds to minimizing the total cost of the packing. This problem arises in a variety of interpretations from computer storage allocation to stock cutting.

This variable sized bin packing problem is NP-Hard, reducing to the classical bin packing problem when only one bin size is permitted. Hence we focus our efforts on efficient approximation algorithms to ensure near-optimal packings.

The classical bin packing problem and many of its variations are of fundamental importance, reflected in the impressive amount of research reported (see [4] for an updated survey). Yet only a few results have appeared concerning this more general problem in which bins need not be a single given size [6], [7], [11]. In particular, it is known from [7] that if the user is allowed to choose any single bin size then any packing algorithm can be employed iteratively to yield a worst-case performance bound arbitrarily close to its bound for the classical problem.

The problem we consider herein appears to be substantially more difficult and our results are more modest. In Section 3, we design two procedures and show they guarantee tight worst-case performance ratios of 2 and 3/2 respectively. Our main result is a proof that a slightly more complex algorithm we name FFDLS, to be described and analyzed in Section 4, never uses more than 4/3 the minimal amount of space. Moreover, this bound is tight.

2. THE MODEL

Let k denote the number of distinct bin sizes available. A list of pieces is specified by $L = \{p_1, p_2, \ldots, p_n\}$. Assume bin and piece sizes are always normalized so that the largest bin has size 1 (and hence the largest piece must have a size no greater than 1). We use $B = \{B_1, B_2, \ldots, B_\ell\}$ to signify the bins used by a heuristic H and

$B^* = \{B_1^*, B_2^*, \ldots, B_m^*\}$ to represent those used by some optimal procedure OPT. $|B_i|$ ($|B_i^*|$) denotes the number of pieces assigned to the i^{th} bin by the heuristic (optimal) packing.

In this setting, we employ $s(p_i)$ to denote the size of the i^{th} piece. We use s similarly to represent bin sizes and a function c to signify the total contents of a bin (naturally $c(B_i) \le s(B_i)$). Since we seek a bound on the space required by an algorithm, we define $H(L) \equiv \sum_{i=1}^{\ell} s(B_i)$ and $OPT(L) \equiv \sum_{i=1}^{m} s(B_i^*)$. We will establish results of the form $H(L) \le R \cdot OPT(L) + C$ where R and C are the asymptotic worst-case performance ratio and the additive constant respectively.

3. THE ALGORITHMS NFL AND FFDLR

Guaranteeing a bound of 2 is very easy. Consider a simple heuristic NFL (for Next Fit using Largest bins only) which packs a bin of size 1 until a piece will not fit, in which case it begins a new bin of size 1 and so on. NFL is clearly of time complexity $O(n)$, linear in the number of pieces which must be packed.

Theorem 1. $NFL(L) < 2\, OPT(L) + 1$ for any list L.

Proof. For $1 \le i < \ell$, $c(B_i) + c(B_{i+1}) > 1$. Therefore $\sum_{i=1}^{\ell} c(B_i) > (\ell-1)/2$ and

$$NFL(L) = (\ell - 1) + 1 < 2 \sum_{i=1}^{\ell} c(B_i) + 1$$

$$= 2 \sum_{i=1}^{m} c(B_i^*) + 1 \le 2 \sum_{i=1}^{m} s(B_i^*) + 1 = 2\, OPT(L) + 1. \square$$

Any packing instance consisting of pieces of size $1/2 + \varepsilon$ and bins of sizes 1 and $1/2 + \varepsilon$ demonstrates that 2 is asymptotic lower bound as well for NFL's performance.

The major stumbling block to improved performance appears to be the lack of any repacking into smaller bins (the interested reader should have no difficulty devising examples to illustrate that initially packing smaller bins does not help). Moreover, algorithms known to be superior to Next Fit for the classical model (see descriptions in [4], [8] or [9]) also fail unless allowed to repack bins. Hence we contemplate FFDLR (for First Fit Decreasing using Largest bins, at end Repack to smallest possible bins). A formal description follows:

```
FFDLR : procedure
begin
    sort L into a non-increasing sequence
    for i ← 1 to n do
        pack $p_i$ into the first bin of size 1 that has room for it
        end
    for i ← 1 to $\ell$ do
        repack contents of $B_i$ into smallest possible empty bin
        end
    end.
```

FFDLR is of time complexity $O(n \log n + \ell \log k)$, comparing favorably with NFL. We now demonstrate that its modest increase in run-time is well worth the effort.

Theorem 2. $FFDLR(L) < (3/2)OPT(L) + 1$ for any list L.

Proof. Suppose otherwise for some L, B and B*. Let i denote the greatest index such that $c(B_i) < (2/3)s(B_i)$ since clearly such a bin must exist. Then $c(B_i) < 2/3$. WLOG, we assume $c(B_i) > 1/2$, else $i = \ell$ and we use the additive constant (+1) to absorb the wasted space in B_ℓ. Similarly, $|B_i| = 1$, else its second piece has size $< 1/3$, $c(B_j) > (2/3)s(B_j)$ for all $j < i$ and the theorem holds with the additive constant absorbing any wasted space in B_i and B_ℓ.

We conclude that there is no bin size in $[c(B_i), 3/4]$ and that each $B_j, j \le i$, contains a piece whose size is $\ge c(B_i)$ and which cannot be packed with any item from $B_{j'}$, $j' > i$. Therefore $FFDLR(L) \le i + (3/2) \sum_{j=i+1}^{\ell} c(B_j) + 1 < (3/2)[(3/4)i + \sum_{j=i+1}^{\ell} c(B_j)] + 1 \le (3/2)OPT(L) + 1$. □

To illustrate that this asymptotic bound of 3/2 is tight, consider any packing instance consisting of an even number of pieces of size $1/3 + \varepsilon$ and bins of sizes 1 and $1/3 + \varepsilon$.

4. THE FFDLS ALGORITHM

Since the single repacking step at the end of FFDLR does not ensure less than 50 percent wasted space, we consider a dynamic shifting of bin contents as the packing is constructed. Focusing our attention on a 4/3 performance bound, we design FFDLS (for First Fit Decreasing using Largest bins, but Shifting as necessary). Formally:

```
FFDLS : procedure
begin
    sort L into a non-increasing sequence
    for $i \leftarrow 1$ to n do
        pack $p_i$ into the first bin of size 1, $B_j$, that has room for it
        if ($B_j$ contains a piece > 1/3 in size)
          then shift, if possible, pieces of $B_j$ to smallest empty bin
            $B_{j'}$ that will hold them where $c(B_j) \ge (3/4)s(B_{j'})$
        end
    for $i \leftarrow 1$ to $\ell$ do
        repack contents of $B_i$ into smallest possible empty bin
        end
end.
```

FFDLS is of time complexity $O(n \log n + n \log k)$. Note that shifting is not performed for bins containing only small pieces (i.e., pieces of size $\le 1/3$). Roughly speaking this is because such a bin, unless it is B_ℓ, must be filled enough over 3/4 to "balance" with bins filled to less than 3/4. To illustrate, suppose we shift anyway and pack 12K pieces, 6K of size .365 and 6K of size .271, using bin sizes 1, .636 and .361. The heuristic requires a space of 3K(1) + 6K(.361) while optimal packing needs only 6K(.636). (FFDLS would not construct an optimal packing either, but would require a space of only 3K(1) + 2K(1) wasting less than 33 percent.)

As described FFDLS does not attempt to pack a piece into any "small" bin already 3/4 full. That is, the first fit property applies only to bins of size 1. In practice we endorse a first fit packing over all partially filled bins, although this modification can have no effect on the worst-case ratio (it cannot make it worse and does not improve the example mentioned at the close of this section).

We now proceed with a proof that FFDLS guarantees packings whose asymptotic space requirements never exceed 4/3 the optimal. The majority of the proof will rely on "weighting function" arguments (the curious reader is referred to [5] for an exposition of this technique).

Theorem 3. $FFDLS(L) < (4/3)OPT(L) + 2$ for any list L.

Proof. Suppose the contrary. Then there exists an L, B and B* such that $FFDLS(L) \geq (4/3)OPT(L) + 2$. WLOG, assume this instance is minimal. That is, k is the minimum number of distinct bin sizes for which the statement of the theorem may not hold and, given k, n is the minimum number of pieces needed for a counter-example. For simplicity assume $s(p_1) \geq s(p_2) \geq \ldots \geq s(p_n)$.

We call a bin B_i "short" if $c(B_i) < (3/4)s(B_i)$. Clearly such a bin exists. We define piece types as follows: p_i is of type X_1 if $s(p_i) > 1/2$, of type X_2 if $1/2 \geq s(p_i) > 1/3$, of type Y otherwise. We shall show later that only pieces of size greater than 1/6 need be considered. In that case, a bin can be short only under very special conditions. If a short bin, B, contains an X_1 piece, a, then B can contain at most one additional piece, b, where $s(b) < (1 - s(a))/2$. Moreover, b must be the only Y piece available for B, since any larger piece would have been used instead of b and any piece no larger would have been packed with a and b. If a short bin, B, contains an X_2 piece as its largest piece, then either B contains two X_2 pieces and nothing else or there are no more X_2 pieces available for B. Except for B_ℓ, a short bin's largest piece cannot be of type Y. We thus allow for three "exceptional" short bins: the last such bin to contain a Y piece, the last to contain an X_2 piece, and B_ℓ. Note that at most two of these may exist in the same packing, and we use the additive constant to absorb their space. Therefore, a non-exceptional short bin must contain either a single X_1 piece or two X_2 pieces. The last short non-exceptional bin will be denoted by BIN in the remainder of the proof.

We divide the analysis into two cases, based on |BIN|. For each we will construct a weighting function w on L. A piece's weight will be dependent on its type and its placement in the FFDLS packing. By the proper definition of w we will show that

$$(3/4)\left(\sum_{i=1}^{\ell} s(B_i) - 2\right) \leq \sum_{i=1}^{\ell} w(B_i) = \sum_{i=1}^{m} w(B_i^*) \leq \sum_{i=1}^{m} s(B_i^*).$$

Case I. $|BIN| = 2$.

It is not difficult to see that $s(p_n) > 1/5$. Suppose otherwise. Let B_i denote the FFDLS bin containing p_n. For $1 \leq j < i$, $c(B_j) \geq (3/4)s(B_j)$. $c(B_i) - s(p_n) > 3/5$, else for $i < j \leq \ell$ either $|B_j| = 1$ or $c(B_j) > (3/4) s(B_j)$. Minimality dictates that p_n affect the packing. Thus there is some bin size b satisfying $(c(B_i) - s(p_n)) + s(p_n) > b > (4/3)(c(B_i) - s(p_n))$ and we derive $s(p_n) > 1/5$.

We next show that every short non-exceptional bin contains two X_2 pieces. The only other possibility is a B_i containing X_1 piece z. Since BIN is not exceptional, the size of its second piece lies in the range $(1/4, 3/8)$. Hence B_i is not short unless z is its only piece and $s(z) > 5/8$. But if this is the case, we can construct a smaller counter-example as follows. Delete z from L. If the optimal bin which had contained z, B_j^*, has no other pieces, halt. Otherwise delete the remaining piece of B_j^* (it can have only one) and z', the X_1 piece packed with it in the FFDLS packing (z' must have been packed before z). Now examine the optimal bin which had contained z' and so on. This procedure halts in a finite number of steps. L has been shortened. Both packings have lost the same number of bins. Every optimal bin deleted has size $> (4/3)(5/8) > 3/4$. Hence we still have a counter-example.

The function w is defined in Table 1. (Pieces from exceptional bins are given a weight of zero.) Note that for any non-exceptional bin B_i, $w(B_i) = (3/4)s(B_i)$. Also, any piece p_i from a non-exceptional bin satisfies $w(p_i) \leq s(p_i)$, unless p_i is an X_2 piece of weight $> 1/3$.

Contents of B_i	Piece Weights (times $s(B_i)$)	Restrictions
X_1	3/4	$s(X_1) > (3/4)s(B_i)$
X_1, X_2	1/2, 1/4	$s(X_1) > 1/2$, $s(X_2) > 1/3$
X_1, Y	5/8, 1/8	$s(X_1) > 5/8$, $s(Y) > 1/5$
X_2	3/4	$s(X_2) \geq (3/4)s(B_i)$
X_2, X_2	3/8, 3/8	$s(X_2) > 1/3$
X_2, X_2, Y	5/16, 5/16, 1/8	$s(X_2) > 1/3$, $s(Y) > 1/5$
Y, Y, Y	1/4	$s(Y) > 1/4$
Y, Y, Y, Y	3/20	$s(Y) \quad 1/5$

Table 1. The function w for Case I.

Therefore the proof of Case I reduces to showing that even if such an X_2 piece is in B_i^*, $w(B_i^*) \leq s(B_i^*)$.

Suppose $|B_i^*| = 1$. Let b denote the single piece of B_i^*. FFDLS cannot pack b as the first piece in a bin, else $w(B_i^*) \le 3/8 < 4/9 < (4/3)s(b) < s(B_i^*)$ since b was not shifted. Let B_j represent the last short bin containing an X_2 piece (such a bin must exist and is exceptional). b cannot have been packed in a bin preceding B_j, else every X_2 piece available for B_j would have been shifted into a bin of size $\le s(B_i^*)$. Neither can b have been packed in a bin following B_j, else such a bin is short contradicting the definition of B_j or has size $< c(B_j) < 3/4$ and $w(b) < (3/8)(3/4) < 1/3 < s(b)$. The only remaining possibility is that b comes from B_j in which case $w(b) = 0$.

Suppose $|B_i^*| = 2$. Suppose the second piece was not shifted into a smaller bin by FFDLS. If this piece is an X_1, then $s(B_i^*) > 5/6$ and $w(B_i^*) \le 1/2 + (3/8)s(B_i^*) < s(B_i^*)$. If this piece is an X_2, then $w(B_i^*) \le 3/8 + \max\{1/4, (3/8)s(B_i^*)\} < s(B_i^*)$ since $s(B_i^*) > 2/3$. It can only be that this piece is of type Y. It is packed by FFDLS into a bin containing only Y pieces, else $w(B_i^*) \le 3/8 + 1/8 < 1/3 + 1/5 < s(B_i^*)$. Thus $w(B_i^*) \le 3/8 + 1/4 < 3/8(2s(X_2) + s(Y)) + 1/4 < s(B_i^*) - (1/4)s(X_2) - (5/8)s(Y) + 1/4 < s(B_i^*) - (1/4)s(X_2) - (5/8)(1 - 2s(X_2)) + 1/4 = s(B_i^*) + s(X_2) - 3/8 < s(B_i^*)$.

Suppose the second piece of B_i^* was shifted when packed in B_j by FFDLS. This piece can only be an X_1 or an X_2. Suppose the piece is an X_1. $s(B_j) < 8/9$. $2s(X_2) > s(B_j)$, else $w(B_i^*) \le w(X_1) + (3/8)(8/9) < s(B_i^*)$. But now $w(B_i^*) \le (3/4)s(B_j) + (3/8)s(B_i^*) < (3/2)s(X_2) + (3/8)s(B_i^*) < (3/4)^2 s(B_i^*) + (3/8)s(B_i^*)$. Suppose the piece is an X_2. The shifted piece is larger than the other, else the other would have been shifted too. $s(B_i^*) > 8/9$, else $w(B_i^*) \le w(X_2) + (3/8)s(B_i^*) \le s(X_2) + 1/3 < s(B_i^*)$. Even so, $w(B_i^*) \le w(X_2) + (3/8)s(B_i^*) \le 3/4 < 8/9 < s(B_i^*)$.

Suppose $|B_i^*| = 3$. $s(B_i^*) > (4/3)(2s(X_2)) > 8/9$, else the X_2 piece was shifted since it cannot be exceptional. B_i^* cannot contain an X_1 nor three X_2 pieces. If it contains two Y pieces, $w(B_i^*) \le 3/8 + 1/2 < 8/9 < s(B_i^*)$. The only remaining possibility is that it contains two X_2 pieces and a Y piece. $w(Y) > 1/8$, else $w(B_i^*) \le 7/8 < 8/9 < s(B_i^*)$. Thus the Y piece was available when the X_2 pieces were packed and $w(Y) \le (1/4)s(B_i^*)$. The smaller X_2 must have been shifted, since otherwise the Y piece would fit implying $w(B_i^*) \le (3/8)s(B_i^*) + 5/16 + (1/4)s(B_i^*) < s(B_i^*)$. It must have been shifted alone, else $w(B_i^*) \le 2(3/8)s(B_i^*) + (1/4)s(B_i^*) = s(B_i^*)$. In fact, it must have been shifted into a bin of size $> 4/9$, else $w(B_i^*) \le (3/8)s(B_i^*) + (3/4)(4/9) + (1/4)s(B_i^*) < s(B_i^*)$. But this means that the larger X_2 would have been shifted alone as well unless it is the second piece of an exceptional bin.

Suppose $|B_i^*| = 4$. It must contain three Y pieces, only one of which may have a weight as large as 1/4. Hence $w(B_i^*) \le 3/8 + 1/4 + 2(3/20) < 1/3 + 3(1/5) < s(B_i^*)$.

Case II. $|BIN| = 1$.

Let B_i denote the FFDLS bin containing p_n. p_n must affect the packing, so $c(B_i) - s(p_n) > 1/2$ and some bin size b satisfies $(c(B_i) - s(p_n)) + s(p_n) > b > (4/3)(c(B_i) - s(p_n))$ implying $s(p_n) > 1/6$.

We now show that there is no FFDLS bin containing a Y piece without an X_1 piece. Suppose otherwise. Then $c(BIN) > 2/3$. Hence we can find a smaller counter-example using the construction described in the previous case.

The function w is defined in Table 2. (Pieces from exceptional bins are again assigned weight zero.) $w(B_i) = (3/4)s(B_i)$ if B_i is not exceptional. $w(p_i) \le s(p_i)$ unless p_i is an X_1 piece of weight > 1/2. If an FFDLS bin contains an X_1 piece and two Y pieces, the smaller Y piece receives weight zero.

Contents of B_i	Piece Weights (times $s(B_i)$)	Restrictions
X_1	3/4	$s(X_1) > 1/2$
X_1, X_2	1/2, 1/4	$s(X_1) > 1/2, s(X_2) > 1/3$
X_1, Y	3/4, 0	$s(X_1) \ge 2/3, s(Y) > 1/6$
	2/3, 1/12	$s(X_1) < 2/3, s(Y) > 1/6$
X_1, Y, Y	2/3, 1/12, 0	$s(X_1) < 2/3, s(Y) > 1/6$
X_2	3/4	$s(X_2) \ge (3/4)s(B_i)$
X_2, X_2	3/8, 3/8	$s(X_2) \ge (3/8)s(B_i)$

Table 2. The function w for Case II.

The proof of Case II reduces to showing that even if such an X_1 is in B_i^*, $w(B_i^*) \le s(B_i^*)$. We first dispose of the possibility that B_i^* contains an X_2. Suppose such is the case. Let B_j represent the FFDLS bin containing the X_1. B_j cannot contain a second piece as large as the X_2, else we delete the pieces of B_j and move the X_2 from B_i^* to the position in the optimal packing formerly occupied by the second piece of B_j, contradicting minimality since $s(B_i^*) > 5/6 > 3/4$. Nor can B_j contain a second piece smaller than the X_2, else FFDLS has packed the X_2 with an earlier X_1 and a similar construction violates minimality. We conclude that FFDLS packs the X_1 alone in B_j, though the X_2 is available and would fit. Hence $w(X_1) \leq s(X_1)$, since the X_1 must have been shifted.

Suppose $|B_i^*| = 1$. Then minimality requires that FFDLS pack the X_1 with another piece. If $s(X_1) \geq 2/3$ then $w(B_i^*) \leq 3/4 < (4/3)(2/3) \leq (4/3)s(X_1) < s(B_i^*)$. If $s(X_1) < 2/3$, then $w(B_i^*) \leq 2/3 = (4/3)(1/2) < (4/3)s(X_1) < s(B_i^*)$.

Suppose $|B_i^*| = 2$. $s(X_1) < 2/3$, else $w(B_i^*) \leq 3/4 + 1/12 = 2/3 + 1/6 < s(B_i^*)$. Let B_j denote the FFDLS bin containing the X_1. $|B_j| > 1$, else $w(B_i^*) \leq (3/4)s(B_j) + 1/12 \leq (3/4)s(B_i^*) + 1/12 < s(B_i^*)$. Let a and b denote the second largest pieces of B_j and B_i^* repsectively. It must be that $a > b$, else either $|B_j| = 2$ and $w(B_i^*) \leq (2/3)s(B_i^*) + 1/12 < s(B_i^*)$ or $|B_j| = 3$ and, lest the X_1 and a be shifted to a bin no larger than $s(B_i^*)$, $w(B_i^*) \leq 2/3 + 1/12 < (4/3)(2/3) < s(B_i^*)$. Suppose b is packed in $B_{j'}$, $j' \leq j$. Then either b is the third piece of $B_{j'}$ or $B_{j'}$ contains an item of size $> 2/3$ since a was not used instead. $w(B_i^*) \leq 2/3 + 0 < s(B_i^*)$. Suppose b is packed in $B_{j'}$, $j < j'$. Let $B_{i'}^*$ denote the optimal bin containing the X_1 piece from $B_{j'}$. $w(B_{i'}^*) \leq (2/3)s(B_i^*) + 2(1/12)$ since the first two pieces of $B_{j'}$ would have been shifted to a bin no larger than $s(B_i^*)$. The weights of these two bins now balance, since $w(B_i^*) + w(B_{i'}^*) \leq 2/3 + (1/12)s(B_i^*) + (2/3)s(B_i^*) + 1/6 < s(B_{i'}^*) + (3/4)s(B_i^*) + 1/6 < s(B_{i'}^*) + s(B_i^*)$.

Suppose $|B_i^*| = 3$. Then $s(X_1) < 2/3$. FFDLS can't pack the X_1 piece by itself, else $w(B_i^*) \leq (3/4)s(B_i^*) + 2/12 < s(B_i^*)$. Thus $w(B_i^*) \leq 2/3 + 2/12 = 1/2 + 2/6 < s(B_i^*)$. □

The careful reader may by now suspect that the additive constant is somewhat pessimistic. We conjecture that a more detailed analysis can reduce it to +1.

To demonstrate the tightness of the 4/3 asymptotic ratio, choose any packing instance with an even number of pieces of size 1/2 and bins of size 1 and 2/3.

5. DIRECTIONS FOR FUTURE RESEARCH

Worst-case results are especially useful here since any attempt at establishing analytical average-case figures will be doubly dependent on the input considered (one probability distribution for bin sizes and a second, possibly distinct, distribution for piece sizes). Nonetheless, we remark that average-case analysis is an active field of research (see for example [12]) and techniques may be brought to bear on this problem. It is possible, we think, that FFDLR surpasses FFDLS in this respect since FFDLS was constructed solely to establish the 4/3 bound.

A natural research goal is to devise an algorithm whose worst-case ratio is less than 4/3. Some heuristic designed along the line of FFDLS may well succeed. We have found packing the largest bins first to be an attractive approach, thus providing at least a partial answer to a question posed in [11].

Variable sized bins could be permitted in other bin packing models such as multidimensional packing [2], dual packing [1] or on-line packing [3] (note that NFL is on-line with respect to both bins and pieces). From a more theoretical standpoint, the problem may even allow an "approximation scheme" (see for example [10]).

REFERENCES

1. Assman, S. F., Johnson, D. S., Kleitman, D. J. and Leung, J. Y-T., "On a dual version of the one-dimensional bin packing problem," to be published.

2. Baker, B. S., Coffman, E. G., Jr. and Rivest, R. L., "Orthogonal packings in two dimensions," SIAM J. Comput. 9 (1980), 846-855.

3. Brown, D. J., "A lower bound for on-line one-dimensional bin packing algorithms," CSL Tech Rpt R-864 (1979), Univ. of Illinois, Urbana, IL.

4. Coffman, E. G., Jr., Garey, M. R., and Johnson, D. S., "Approximation algorithms for bin-packing -- an updated survey," in preparation.

5. Coffman, E. G., Jr., "An introduction to proof techniques for packing and sequencing algorithms," in Deterministic and Stochastic Scheduling, M.A.H. Dempster, et al. (eds.), (1982), 245-270, Reidel Publishing Co., Amsterdam.

6. Friesen, D. K. and Langston, M. A., "Bounds for MULTIFIT scheduling on uniform processors," SIAM J. Comput. 12 (1983), 60-70.

7. Friesen, D. K. and Langston, M. A., "A storage-size selection problem," to be published.

8. Johnson, D. S., Demers, A., Ullman, J. D., Garey, M. R., and Graham, R. L., "Worst-case performance bounds for simple one-dimensional packing algorithms," SIAM J. Comput. 3 (1974), 299-325.

9. Johnson, D. S., "Fast algorithms for bin packing," J. Comput. Syst. Sci. 8 (1974), 272-314.

10. Karmarkar, N. and Karp, R. M., "An efficient approximation scheme for one-dimensional bin packing problem," Proc. 23rd Annual Symp. on Foundations of Computer Science, IEEE Computer Soc., 1982.

11. Langston, M. A., "Performance of heuristics for a computer resource allocation problem," SIAM J. Alg. Disc. Meth., to appear.

12. Ong, H. L., Magazine, M. J. and Wee, T. S., "Probabilistic analysis of bin packing heuristics," Operations Res., to appear.

LINEAR TIME UNIVERSAL COMPRESSION TECHNIQUES FOR DITHERED IMAGES BASED ON PATTERN MATCHING (Extended Abstract)*

ALBERTO APOSTOLICO
Computer Sciences Department, Purdue University
West Lafayette, Indiana 47907 - and -
I.S.I., University of Salerno, I-84100 Salerno, Italy

ERNESTO GUERRIERI
I.P.L., Rensselaer Polytechnic Institute
Troy, N.Y. 12181

ABSTRACT

Simple combinatorial properties of dithered discrete signals are derived that enable to set up variants of the Lempel-Ziv universal compression technique that take linear time by brute force and yet can outperform the compaction achieved by purer - but computationally more involved - variants.

1. INTRODUCTION AND SUMMARY

Dither transformations [JJ] techniques for multilevel pictures consist of thresholding each original pixel with a suitable position dependent integer value chosen from a finite set in a systematic way. The idea is that of faking regions of more or less uniform intensity by different spatial distributions of dots of uniform sizes.

Efficient techniques for the compression of dithered images are much needed. However, the nature of the dithering process offsets such popular approaches as the encoding of run lengths [GO]. On the other hand, predictive encodings are fairly sensitive to the statistics of the source, whereas [HO] *"the problem of how to choose the optimum prediction function and the size of the area on which prediction is based remain unsolved"*. Other ad-hoc methods convey nonnegligible precomputations and sometimes encodings (and corresponding decodings) that are not strictly sequential. In addition, some are information lossy and, because of a strongly heuristic nature, share with the encoding of run lengths an unpalatable lack of theoretically grounded *universal* estimation of performance. This paper originated from an attempt to apply to dithered images the various incarnations of the Lempel-Ziv universal compression technique [LZ,ZL,RP,ZI,ZL1,SZ1,SZ]. Due to the somewhat subtle interplay existing among the opulence of the variation applied, its absolute time requirements and the average size of pointers in the output encodings, experiments with sizeable yet realistic buffers, would not yield exciting results in at least one such respect.

We show here that dithered images and signals are susceptible of a characterization of algebraic flavor and manifold interest. Such characterization exploits the notions of periodicities and repetitions for a string [AP]. The detection of such regularities is at the heart of the Lempel-Ziv methods, so that they happen to take some role in the afore mentioned interplay. The set of repetitions in a generic string can be quite rich in general, attaining the bound of $O(n \log n)$ [CH] for a string of length n: however, we show that plausible limitations can be drawn from the dithering assumption. More precisely, we show that:

---

* Work by the first author was supported in part by the Italian National Council for Research and by the Italian Ministry of Education Scientific Program. It was partly carried out during his visit with the ECSE Dept. of Rensselaer Polytechnic Institute, where additional support was provided by a grant by IBM Corporation. Further support was also provided by NATO under Research Grant 039.82. Work by the second author was supported by the Image Processing Laboratory of RPI.

- for a broad variety of dithered discrete signals, the number of such repetitions can be expected to be $O(n \log p)$, p being the size of the *dither vector (or matrix)*. (Usually it is $p << n$).
- a brute force strategy that guarantees to find all such repetitions can be set up to work in time $O(pn)$ and linear space.
- a class of parsing methods, based on the above, can be devised that fall within the Lempel-Ziv taxonomy and yet enjoy the following nice features: for reasonable sizes of the buffer, on one side the vocabulary bookkeeping is sharply reduced with respect to the *pure* Lempel-Ziv strategy [LZ]; at the same time, the number of phrases in the parse is close to that obtained by the former and, due to the smaller average pointer size, the compression achieved may be even better.

We suffragate the above with results from preliminary experiments.

2. A TAXONOMY OF UNIVERSAL DATA COMPRESSION TECHNIQUES

We shall deal with *strings* of *symbols* from a finite alphabet I. If x is a string, $|x|$ will denote the *length* (i.e., the number of symbols) of x. Sometimes we will implicitly assume that it is $|x| = n$. Letting \$ be a special symbol not belonging to I, we will assume that $x\$$ is stored into an array $x[i:n+1]$. For each i in the set $P=\{1,2,...n+1\}$ of *positions* of $x\$$, $\mathrm{suf}_i = x[i:n+1]$ shall denote the *ith suffix* of $x\$$.

Let $Q = \{i_1, i_2, ..., i_m\}$ be an ordered subset of P, and assume that all suf_{i_j} $(j=1,2,...,m)$ have been inserted into a digital search tree that shall be denoted by T_Q. We remark that, in particular T_P coincides with the suffix tree [MC] associated with x. For any possible selection of Q, T_Q has exactly m leaves and at most $m-1$ internal nodes.

It is convenient to think of T_Q as obtained by starting with an empty tree T_0, and then by inserting at the *jth* step $\{j=1,2,...,m)$ suf_{i_j}, $i_j \epsilon Q$, in the tree T_{j-1} to produce the *jth* update of the tree T_j. We remark that, since \$ does not belong in I, then for each $i_j \epsilon P$ we are able to write:

$$\mathrm{suf}_{i_j} = \mathrm{head}_j * \mathrm{b} * \mathrm{tail}_j * \$ \quad \text{or} \quad \mathrm{suf}_{i_j} = \mathrm{head}_j * \$$$

where * is the concatenation operation, b is a symbol of I, and head_j is the longest prefix of suf_{i_j} that is also a prefix of suf_{i_f} for some $f < j$, if such a prefix exists, or otherwise $\mathrm{head} = \lambda$, i.e., the *empty word*. It follows from this observation that there is an internal vertex α in T_j $(j=1,2,...,m)$ such that the path from the root of T_j to α *spells out* the symbols of head_j. In general, a vertex α of T_Q shall be called the *locus* of any substring of x the path associated with which terminates at a point where α is the closest (leafward) vertex [MC].

A *generative process* for x is a set of rules to retrieve the set Q from P, on the basis of an uniform criterion that identifies, for each suffix of x with starting position $i_j \epsilon Q$, the associated *jth reproduction* rep_j of x and its strictly related *production* prod_j. The exact nature of rep_j depends on the particular generative process chosen. In all of our cases, however, rep_j will coincide with a suitable prefix of a suffix suf_{i_f}, with $i_f \epsilon Q$ and $f < j$; in addition, the following definition will always hold for prod_j:

$$\mathrm{prod}_j = \mathrm{rep}_j * x[i_j + |\mathrm{rep}_j|]$$

Thus prod_j is fully individuated by setting some suitable pointer(s) to the previous suffix and by providing the (possibly innovative) terminal symbol. We call this information the *identifier* for prod_j, and denote it by $id(j)$.

For each type $(A, B, C, ...)$ of reproduction defined, the *<type>-parse* of x, denoted $type-P(x)$ is the (unique) decomposition of x in terms of those productions that are pinpointed through the greedy left to right scanning of the symbols of x. The production of x that is selected by actuating the $h-th$ step in this process represents the $h-th$ *phrase* in the parse. Since each phrase is also a production, then we can associate with the parse of x its *translation* $\sigma(x)$, defined as the concatenation of the identifiers for the reproductions that are also phrases in the parse. Notice that, according to our definition, rep_j may or may not be a prefix of head_j and that prod_j might have a locus in T_j. In any case, all the productions of x are actually found in a tree that can be obtained from T_Q by a suitable pruning of some of its subtrees. In the following, we shall assume that auxiliary nodes are ideally inserted in T_Q, whenever needed, as endmarkers for those paths from the root that spell out the reproductions of x. It goes without saying that the number of such auxiliary nodes cannot exceed m, whence they do not affect the space requirements for T_Q.

We now characterize a class of parses into which all of the Lempel-Seery-Ziv strategies can be casted, and describe their associated translations along with some of the economical encodings that can be devised for these latter. The listing of the procedure *parse* below is largely self explanatory. We shall not belabor the implementation of the insertion of each newcomer suffix into the tree, which we concisely describe as an UNION operation. It should be mentioned, however, that this task calls for the recognition of the locus of head_j at each step: we assume that this operation is accompanied with the identification of the current (re)production via the auxiliary function Lprefix, and by the insertion of an auxiliary endmarker node whenever appropriate. Finally, it should be remarked that the nature of the identifiers for productions of x depend on the particular scheme chosen, with the noteworthy exception of the first one: indeed, this latter coincides, for all possible parses, with the plain symbol $x[1]$. As our exposition proceeds, the reader shall realize that the identifiers with their associated *encodings* constitute in fact a crucial issue and the actual "leitmotiv" of this paper.

```
procedure parse $(x, r)$
## produces $\sigma(x)$ from inputs $x$ and the integer parameter $r$ ##
begin $i:=1$; $j:=1$; $h:=1$; $T_j:=T_0 \cup \{\text{suf}_i\}$;
        $\text{phrase}_h:=\text{prod}_j:=x[i]$;$\sigma:=id(j):=<x[i]>$;
while $i<n$ do ## produce next phrase ##
    begin $i:=i+1$; $j:=j+1$; $h:=h+1$;
    $T_j := T_{j-1} \cup \{\text{suf}_i\}$;
    $\text{phrase}_h:=\text{prod}_j:=L\text{prefix}(\text{suf}_i)$;
    $\sigma=\sigma * id(j)$;
    if $i+1|\text{rep}_j|<n$ then
        ## generate intermediate (re)productions ##
        begin if $|\text{prod}_j|<2r$
                then
                    for $k:=i+1$ to $i+|\text{rep}_j|$ do
                        begin $j:=j+1$; $T_j:=T_{j-1} \cup \{\text{suf}_k\}$;
                        $\text{prod}_j:=L\text{prefix}(\text{suf}_k)$
                        end
                else for $k:=i+1$ to $i+r+1$
                    and for $k:=i+|\text{rep}_j|-r$ to $i+|\text{rep}_j|$
                        begin $j:=j+1$;$T_j:=t_{j-1} \cup \{\text{suf}_k\}$;
                        end
        end
    $i:=i+|\text{rep}_j|$
    end
    $\sigma(x):=\sigma$
end.
```

With the aid of a short running example, we now single out the instantiations of *parse* that result from some of the possible selections for the parameter r and from the somewhat related semantics for the operator Lprefix. Our sample string x is as follows:

x = 11111111111111111110111011101110111010101010001000100

Henceforth we also assume to deal only with binary strings. Moreover, log will denote the logarithm to the base 2[z] denotes the largest integer smaller than z, c is used for the number of phrases in a parse. Of the schemes described below, the $A/B/C/D$-parses are from [SZ1,SZ]: their unifying feature is that it is set $r=0$, whence it results $j=h$ at all times during *parse*. For the first two schemes, it is shown that by a slightly more complicated construction the last bit of each phrase need not be transmitted (the reader is referred to the quoted references for details). The E-parse is from [LZ,RP]: basically, it arises by allowing an unlimited number of intermediate insertions of reproductions in the tree (i.e., by setting $r>n$, so that it is now $j=i$ throughout *parse*, whereas $h>j$ for a string of more than two symbols. Our encoding conventions follow closely those in [SZ,SZ1]. Indeed, more sophisticated encodings such as those based on *universal representations of the integers* [EL,RP], proved to cause a worse compaction for the buffer sizes to be used in this application.

The A-parse is characterized as follows:

Lprefix(suf$_i$) - coincides with the longest prefix of suf$_i$ that matches some past production (=phrase), extended by concatenation of the next symbol of suf$_i$.

Example: $A-P(x)$ = 1-11-111-1111-11111-11110-1110-11101-110-111010-10-101-0-00-100-01-00$

c = 17

$|id(j)| \simeq [\log j] = [\log h]$ bits [SZ1] (i.e., roughly the bits needed to identify one among $h-1$ previous phrases plus the empty phrase λ.)

The B-parse is as follows:

Lprefix(suf$_i$) - is given by the longest prefix of suf$_i$ that matches the concatenation of two past phrases followed by the terminal symbol as above, or else it is as per scheme A if no such pair of phrases exists.

Example: $B-P(x)$ = 1-11-1111-1111111-11111-0-1110-11101-110-111010-10-101-00-010-001-00$

c = 16

$|id(j)| \simeq [\log(3j)] = [\log(3h)]$ [SZ] (roughly, the current phrase is identified by selecting one of the h possible simple phrases, plus the $h-2$ pairs followed by an incoming 1, plus as many pairs followed by a 0).

The C-parse and the B-parse are closely related. For the first one we have:

Lprefix(suf$_i$) - is chosen as the longest concatenation of *past* phrases, ending perhaps in a prefix of a past phrase, followed by the new symbol as above.

Example: $C-P(x)$ = 1-11-1111-11111111-1110-11101-110-1110111010-10-101-0-00-10001-00$

c = 14

$|id(j)| \simeq [\log h] + [\log i] + 1$ ($[\log h]$ bits are needed to identify the first past phrase, $[\log i]$ bits contain the length of the current phrase and the last bit is needed for the terminal symbol).

In the D-parse, we waive the requirement that the copying process be terminated during some *past* phrase, i.e., we have now:

$\text{rep}_j = \text{head}_j$

Example: $D-P(x)$ = 1-111111111111111110-1110-111011101110-10-1010100-0-100-0100$

c = 9

$|\text{id}(j)|\ \tilde{=}\ [\log h] + [\log(n-i+1)] + 1$ (this has an interpretation similar to that of scheme C, except that the length of the current phrase can exceed the value of i).

The L suffix in the E-parse is exactly the same as for the D-parse except that it is now $r=n$. It follows that it also $\text{rep}_j = \text{rep}_i = \text{head}_i$.

Example: $E-P(x)$ = 1-111111111111111110-111011101110111010-1010100-01000-100$

c = 6

$|\text{id}(j)|\ \tilde{=}\ [\log i] + [\log(n-i+1)] + 1$ (the copying process may now start at any past position).

At first sight, one might be inclined to judge that scheme E is the best one, since the value of c is lowest for this scheme. One important observation, however, is that as the value of c decreases the size of identifiers show a corresponding tendency to increase. Thus a conclusion cannot be drawn until the actual *encoding* for each scheme, defined as the length of $\sigma(x)$ in bits when the corresponding $|\text{id}(j)|$'s are computed as above, is considered.

It is readily seen that the instantiations $A-D$ of *parse* trivially run in linear time. Indeed, the number of character matches performed during the definition of each phrase in any such parse add up to n, i.e., the sum of the characters in each individual phrase. One the other hand, a brute force implementation of the E-parse yields a quadratic worst case performance. Although, there are clever implementations of the E-parse that work in linear time [RP] their associated programming and storage overhead would hardly fit the small, dedicated processors that could be currently considered as viable hosts for the application at hand. In the following, we study simple combinatorial properties that are enjoyed by sequences resulting from a dithering process. This shall legitimate intermediate instantiations of *parse* that are obtained by setting r to a small (i.e., negligible with respect to n) value. Thus, on one side, the procedure will run in practically linear time even if we let the suffix insertion operation to be implemented via a leafbound search from the root of the tree. On the other, $j(\log j)$ is kept in this way more close to h $(\log h)$ than to $i(\log i)$.

3. PERIODICITY IN STRINGS AND SOME RELATED CONCEPTS

A few more definitions are in order. We say that p is a *period* of x if $x[i]=x[i+p]$ $(i=1,2,...,n-p)$. Conversely, x is *periodic* if x has a period $p \leq |x|/2$. Let w be a substring of x. We say that w is *primitive* if setting $w=u^k$ for some $k>1$ and $u \neq \lambda$ implies $k=1$ and $u=w$. A string w is a square if it can be put into the form $w=u^2=uu$. We remark that if w is a square then there is a square prefix vv of w with v primitive. The string w is *strongly primitive* or *square-free** if there is no square in w.

---

* It is impossible to write a square-free word of length larger than 7 on a 2-symbols alphabet. However, such indefinitely long strings exist for $|I| > 3!$ [TH].

Useful notions are also those of *repetition* and *maximal repetition*. A *repetition*, is a positioned periodic substring of x the periodicity of which cannot be extended to the right. A *maximal repetition* is a repetition that in addition to the above cannot be extended to the left. Any repetition can be thought of as produced by the contiguity or overlap of two identical substrings of x.

Lemma 0: $x[i:m]$ is a repetition of x iff there is a substring w of x in the form $w=(st)^k s$, with st primitive and $k \geq 0$ such that: (a) there are occurrences of w in x with starting positions i and $i+|st|$, and (b) it is $x_{m+1} \neq x_{m+1-|st|}$, or $m+1 > |x|$.

It is straightforward to see that a repetition is uniquely identified by its starting position i, its period p and its duration or length L, as follows. Since a repetition can be always put into the form $(st)^k s$ with $k>1$, st primitive and s nonempty, it is sufficient to specify, with its starting position i in x, the value of the period $p=|st|$ and that of $L=|(st)^k s|$. In the following, we will refer to a repetition as to a triplet $R(i,p,L)$, with obvious meanings.

Optimal algorithms are available to spot all repetitions in an assigned string x [CR,AP,ML]. For our purposes, we shall describe a suboptimal repetition finder that can be easily casted into the brute force construction of the suffix tree associated with a generic input string x.

Assume that suf_i be inserted in T_{i-1} by tracing a leafward path from the root of this tree down to the point where a mismatch occurs. Let β be the closest (leafward) node at this point. If the mismatch occurs in the middle of an arc, then a new node has to be inserted to be the locus of head_i in T_i. Let α be this node, and assume also that, whenever a node is visited during the insertion of a newcomer suffix, a special field LAST is updated to contain the starting position of this suffix in x. Finally, for any vertex v, let $W(v)$ be the substring of x that corresponds to the path from the root of the tree to this vertex.

Lemma 1: $R(i,(i-j),(i-j)+|W(\alpha)|)$ is a repetition iff $|W(\alpha)| \geq (i-j)$ and i and j are consecutive leaves in the subtree of T_i rooted at α.

Lemma 1 [AP] is a direct consequence of the following well known [SH]:

Lemma 2: *(periodicity lemma)*: if w has periods p and q and $|w| \geq p+q$ then w has period g.c.d.(p,q).

Corollary: $R(i,(i-j),(i-j)+|W(\alpha)|)$ is a repetition iff $|W(\alpha)| \geq (i-j)$ and letting β to be the locus of head_i in T_{i-1}, it is, in T_{i-1}, $\mathrm{LAST}(\beta)=j$.

Thus when the insertion suf_i reaches β a repetition can be detected simply by comparing $[i\text{-LAST}(\beta)]$ with the value of $|\mathrm{head}_i|$. We will now characterize a class of repetitions that can be recognized in less than quadratic time by brute force.

4. DITHER TRANSFORMS, QUASISIMPLE REPETITIONS AND PSEUDORUNS

We say that a string x is *root-bounded of degree p* if there is no repetition in x the root of which has length larger than p.

Let $z=z_1,z_2,..z_n$ be a string of integers in the range $[0,r]$, and let $V=\{v_1,v_2,\cdots v_k\}$ $(k<n)$ be a suitable vector of thresholds (the *dither vector*), where each $v_i \in [0,r]$. We shall study the binary string x which is obtained from z by the following nonlinear transformation.

$$\text{for each } i \in [1,n],\ x[i] = \begin{cases} 1, & \text{if } z_i \geq v_{i(mod\,k)} \\ 0, & \text{otherwise} \end{cases}$$

Some such transformation is often used to generate binary object images from multilevel sources, so as to fake the intensities of the original pixels by different concentrations of uniformly sized binary dots [JJ].

Let V' be the set obtained by sorting the elements of V in ascending order and by eliminating then possible repeated terms. We now split z into consecutive *pseudoruns*, these latter being defined as follows. A *pseudorun* is any longest substring of elements of z, such that the generic element z_i in the substring observes: $z_i \in [v', v''[$, with v' and v'' two consecutive elements of V'. Notice that, in particular, a run of identical elements in z will be a substring of a pseudorun for all possible choices of the vector V. The interest of pseudoruns lies in the following simple theorem.

Theorem 1: If $z_1 \cdots z_m$ is a pseudorun, then $x[1{:}m]$ is root bounded of degree k.

Proof: The assertion is obviously true for $k > m/2$. Let then be $k \leq m/2$. Then by the definition of dithering, $x[1{:}m]$ has a period q at most equal to k. Assume that there is also a repetition $R(j, p, L)$ in x, with $p > k$. Since it must be $L \geq 2p > p+k$, we have then that there is a substring of length $p+k$ where two periodicities (the one due to R and that caused by the dithering) coexist, a contradiction.

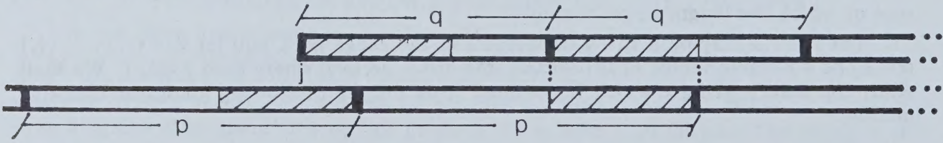
The discussion in the previous section suggests that intercepting quickly the maximal repetitions present in our strings might prove highly rewarding. The study of maximal repetitions can give interesting clues as for their efficient detection. If $R(i, p, L)$ and $R(j, q, L')$ are two maximal repetitions, we shall say that they *chain* if they share at least one position on x, and either it is $j < i$ and $j+L' > i+L$ or else it is $j < i$ and $j+L' < i+L$. We leave it as an easy exercise for the reader to show that, in force of the periodicity lemma, we have:

Lemma 3: Two chaining (maximal) repetitions of x as above can only overlap on a number of consecutive positions of x smaller than $p+q$.

The reader might find it interesting to check that the maximal repetitions in our sample string are either completely disjoint or chaining. In addition, if a repetition is embedded into another, then the former has mostly period 1. Let us say that a repetition is *simple* if it does not embed any other repetition, and that it is *quasisimple* if it might embed some repetitions of unitary period but no other repetitions. The following lemma characterizes simple repetitions.

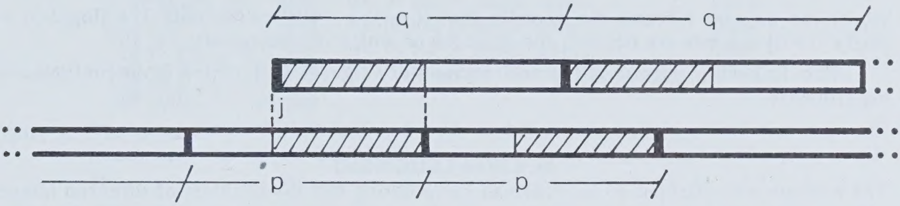
Lemma 4: Let $R(i, p, L)$ and $R(j, q, L')$ $(i < j)$ be two simple chaining repetitions of (primitive) roots u and v, respectively. Then it is also $j \geq i+L-p$ or $j \geq i+L-q$, or both.

Proof: It is easily seen that if $p = q$ then $x[j{:}j+p-1]$ is not a primitive word, which contradicts the assumption that it is the root of a repetition. Hence, it must be $p \neq q$. Let then $r = \max(p, q)$ and assume that $j < i+L-p$. If it is $r = p$, then a situation such as depicted below is encountered.



I.e., since $i+L > j+q$, then there is a nonempty suffix w of u that is also a prefix of v. But it is also $j < i+L-p$ by hypothesis, which implies that there are two occurrences of w in x that overlap on at least one position

j. On the other hand, if $r=q$, then by analog reasoning we derive that there are two occurrences of a suffix of u that overlap at least on the position $j+q$, as shown below.

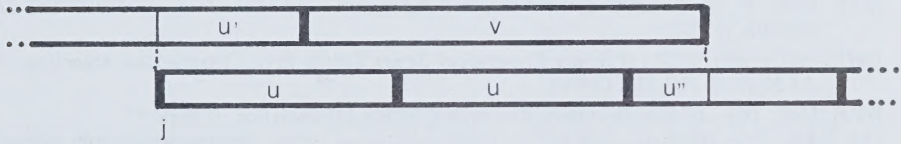


Lemma 4 can be rephrased by saying that two consecutive simple repetitions of x of periods p and q either are disjoint or they overlap over a number of positions of x not larger than $\max(p,q)$.

Lemma 5: Let $R(i,p,L)$ and $R(j,q,L')$ be two chaining quasisimple repetitions. If $j+2q<j+L$ then it must be $p>2q$.

Thus we can have at most $\log p$ chaining quasisimple repetitions in a row.

Proof: Assuming $j+2q<i+L$, it cannot be $p<q$, in force of the periodicity lemma. Also, the assertion is obviously true if it is $j \geq i+L-1$. Let us assume that it is $j<i+L-p$. A situation such as that depicted below is encountered.



i.e., there is a suffix of the (cyclic permutation) of the root v of $R(i,p,L)$ that has the form uu', where u is the root of $R(j,p,L')$. At the same time, there is also a suffix of v that equals a prefix u'' of u. Now it must be $|u''|>|u'|$: otherwise, two overlapping occurrences of u would generate a repetition and, since $R(i,p,L)$ is quasisimple, such repetition can only be in the form $b^{|u|+1}$, meaning that u is not primitive. If $|u''|>|u'|$, from the fact that $2|u|+|u''|=|v|+|u'|$, we derive that $|v|=2|u|+|u''|-|u'|$. Since both u'' and u' are not empty by hypothesis, we have that $|v|>2|u|$.

5. CONCLUSION

The analysis of the preceeding sections shows that nearly all maximal repetitions of a dithered signal are implicitly intercepted by the instantiation of *parse* where it is $r=k$, i.e., where the number of intermediate insertions equals the size of the dither vector. Values of r less than k can also be expected to yield satisfactory results: moreover, if k is a constant independent of n, all such instantiations of parse run trivially in linear time.

At the same time, id(j) has now a cheaper format. Indeed, with the notations of sec.2 and letting m be the past position at which the copying process for rep$_j$ starts, id(j)

may be encoded as follows. If $(i-m)>r$, then:

$$[\mathrm{id}(j)] = 1 + [\log(j)] + [\log(n-i+1)] + 1$$

where the first bit informs the decoder that $(i+m)>r$. Otherwise, only $1 + [\log(2r)] + [\log(n-i+1)] + 1$ bits are needed, the meaning of which is transparent.

The following page displays self explanatory figures and tables from preliminary experiments.

ACKNOWLEDGMENT

The authors are indebted to H. Freeman for pointing out the problem of dithered image compression and for granting them use of the computing facilities of the IPL to carry out the experiments.

REFERENCES

[LZ] Lempel, A. and Ziv, J., *On the Complexity of Finite Sequences,* IEEE TIT *22,* 1, 75-81 (1976).

[ZL] Ziv, J. and Lempel, A., *A Universal Algorithm for Sequential Data Compression,* IEEE TIT, *23,* 3, 337-343, (1977).

[RP] Rodeh, M., Pratt, V.R. and Even, S., *Linear Algorithms for Data Compression via String Matching,* JACM *28,* 16-24, (1981).

[JJ] Judice, C.N., Jarvis, J.F., and Ninke, W.H., *Using Ordered Dither to Display Continuous Tone Pictures on a AC Plasma Panel,* Proc. SID, *15,* 4, 161-167, (1974).

[GO] Golomb, S.W., *Run-Length Encodings,* IEEE TIT *12,* 4, 399-401, (1966).

[EL] Elias, P., *Universal Codeword Sets and Representations of the Integers,* IEEE TIT *21,* 194-203, (1975).

[MC] McCreight, E.H., *A Space Economical Space Suffix Tree Construction Algorithm,* J. ACM *23,* 2, 262-276, (1976).

[HO] Hou, H.S., *Digital Document Processing,* Wiley Interscience, (1983).

[ZI] Ziv, J., *Coding Theorems for Individual Sequences,* IEEE, TIT *24,* 4, 405-413, (1978).

[ZL1] Ziv, J. and Lempel, A., *Compression of Individual Sequences via Variable Length Encodings,* IEEE, TIT *24,* 5, 530-536, (1978).

[SZ1] Seery, J.B. and Ziv, J., *A Universal Data Compression Algorithm: Description and Preliminary Results,* Bell Labs., TM 77-1212-6/77-1217-6, (1977).

[SZ] Seery, J.B. and Ziv, J., *Further Results on Universal Data Compression,* Bell Labs., TM 78-1212-8/78-1217-11, (1978).

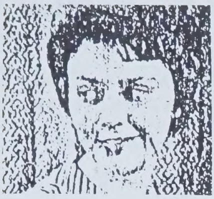
[CR] Crochemore, M., *An Optimal Algorithm for Computing the Repetitions in a Word,* Inf. Proc. Letters *12,* 5, 244-250, (1981).

[ML] Main, M.G. and Lorentz, R.J., *An $O(n \log n)$ Algorithm for Finding All Repetitions in a String,* The Journal of Algorithms, (to appear).

[AP] Apostolico, A. and Preparata, F.P., *Optimal Off Line Detection of Repetitions in a String,* Theoretical Comp. Sci., *22,* 297-315, (1983).

[TH] Thue, A., *Uber Die Gengenseitige Lage Gleicher Teile Gewisser Zeichenreichen,* Skr. Vio. Kristiana I. Mat. Naturv. Klasse 1, 7-67, (1912).

[SH] Lentin, A. and Schutzemberger, M.P., *A Combinatorial Problem in the Theory of Free Monoids in:* Combinatorial Mathematics and its Applications, (Univ. of North Carolina Press, N.C., 1969) 128-144.



No. of lines	:	512
No. of bits in image (total)	:	262144
No. of bits in image (zero)	:	125547
No. of bits in image (one)	:	136597
F.O. Entropy of the entire image	:	0.9987

Figure 1. The input image 'Karen' (dithered with an 8x8 matrix) and its salient features. All parses were carried out considering one line at a time, i.e., with a buffer size of 512 bits.

PARSE		A	B	C	D
Total no. of phrases generated	:	35568	25175	18242	14460
Total bit count in translation	:	186114	162372	189804	165167
Compaction ratio of image (mean)	:	0.7086	0.6182	0.7226	0.6288
Compaction ratio of image (variance)	:	0.0375	0.0794	0.0852	0.1257
Minimum line compaction ratio	:	0.2520	0.1582	0.1055	0.1582
Lines left uncompressed	:	8	242	24	202

Table 1. The Seery-Ziv Taxonomy of parses for the image of Figure 1. C and D parses yield a worse compression, even though the number of phrases decreases. This is to be charged to larger pointer sizes.

Value of *r*		∞	1	4	8
Total no. of phrases generated	:	9979	12739	10704	10273
No. of nodes in parse tree (mean)	:	970.1832	135.5575	306.5555	466.0214
No of nodes in parse tree (variance)	:	20328.9102	6414.3193	33735.0703	70268.2813
No. of nodes in parse tree (min.)	:	4	4	4	4
No. of nodes in parse tree (max.)	:	1024	299	633	633
Total bit count in translation	:	150449	164620	148156	146689
Compaction ratio of image (mean)	:	0.5728	0.6268	0.5641	0.5585
Compaction ratio of image (variance)	:	0.1093	0.1170	0.1060	0.1018
Minimum line compaction ratio	:	0.0273	0.0293	0.0332	0.0352
Lines left uncompressed	:	82	163	65	52

Table 2. Parses of the same figure according to the 'pure' Lempel and Ziv strategy (first column) and with three small values of *r*. Setting $r = 4$ already scores a better compression than that of column 1, even though the number of phrases is slightly larger.

MULTI-RESOURCE EFFICIENT ALGORITHMS FOR INTEGER MULTIPLICATION*

DAVID A. CARLSON

Dept. of Electrical and Computer Engineering, University of
Massachusetts, Amherst, Massachusetts 01003

I Introduction

The multiplication of two integers is a fundamental computational problem, which, due to its obvious importance, has been the subject of a large amount of research concerning efficient implementations. The fastest known algorithm for performing n-bit integer multiplication is due to Schonhage and Strassen [9], and requires 0(n log n loglog n) bit operations. While time forms a major part of the cost of computing, the resource of storage space should also be considered in an analysis of the computational complexity of a problem such as integer multiplication. Of interest are the possible tradeoff relationships that exist between these two resources, and algorithms that make efficient use of both time and space. In this paper, we address these issues by developing algorithms for integer multiplication that illustrate the time-space exchanges inherent in the problem. With respect to their simultaneous time and space requirements, the algorithms are shown to be optimal over a restricted set, and close to optimal over the entire set of algorithms implementing integer multiplication.

The possible tradeoff between time and space for a straight-line computation can be modeled by playing a well known pebble game on a directed acyclic graph whose vertices and edges represent operations and argument assignments of the computation, respectively. Pebbles are used to model the temporary storage used in the computation, and they are moved from input nodes to output nodes according to the following rules:

1. A pebble may be placed on an input node at any time,
2. A pebble may be removed from an output node at any time, and
3. noninput nodes may be pebbled only when all their predecessors have pebbles.

---

* Work supported in part by NSF grant ECS-$\overset{82}{\wedge}$04894

It is easy to see that the maximum number of pebbles used during the pebbling process is a measure of the computation's space requirements, and time is reflected in the number of pebble placements made on the graph (applications of rules 1 and 3). When the amount of available space is decreased, certain nodes of the graph may have to be 'recomputed', since there are no longer enough pebbles to hold the values of intermediate computations. In such cases, a decrease in space requires an increase in time and vice-versa, and it is these time-space exchanges that have been the subject of a large amount of research concerning the pebble game.

Many algebraic computational problems can be solved using methods that are oblivious in nature, i.e. in an algorithmic implementation there are no decision-making steps dependent on actual input values. The pebble game provides a means of studying the simultaneous time and space requirements of these problems. The results obtained are usually expressed as lower bounds on the product of time and space, and in certain cases, are complemented by pebbling strategies that achieve or come close to achieving the known lower limits on a problem's simultaneous time and space requirements. A typical pebble game analysis for an n input computation (graph) shows that when S pebbles are used, $\Omega(n)$ placements on inputs must be made for each set of $O(S)$ outputs pebbled. This is due to the richness in paths between inputs and outputs of the graph, which is a direct consequence of the data dependencies inherent in the computation being performed. The above analysis yields a lower bound of $T = \Omega(n(n/S))$, i.e. $TS = \Omega(n^2)$. Such an analysis can be made on the graph associated with a specific algorithm: different pebbling strategies lead to a class of algorithms for which the above analysis provides lower limits on achievable time and space. It can also be made for a specific problem whose inherent data dependencies imply that a graph representing an arbitrary algorithmic solution has certain connectivity properties between inputs and outputs.

Using the above outlined methods, the time-space tradeoff characteristics of a number of algebraic computational problems have been studied. Savage and Swamy [7] obtained the result $TS = \Theta(n^2)$ for the graph representing an n-input fast Fourier transform, and Tompa [8] extended the lower bound to the problem of computing a discrete Fourier

transform by considering the connectivity properties of the graph associated with any algorithmic implementation. Tompa also obtained results for other problems including straight-line merging and polynomial multiplication using the same techniques, and JaJa [5] has applied them to matrix inversion, obtaining
$TS = \Omega(n^4)$. Carlson [2, 3] investigated algorithms for convolution based on fast Fourier transforms, and showed that $T = \theta(n^3/S^2 + n^2\log n/S)$. The upper bounds imply the existence of time and space efficient algorithms for the multiplication of two polynomials that are optimal over all FFT based methods.

Another approach to deriving lower bounds on the simultaneous time and space requirements of a problem is due to Grigoryev [4], and can be characterized by studying the size of the range of function values associated with a boolean function. The method does not explicitly involve pebbling, but the concepts involved in deriving lower bounds are very similar to pebbling techniques. Grigoryev obtained the bounds $TS = \Omega(n^2)$ for boolean polynomial multiplication and $TS = \Omega(n^3)$ for boolean matrix multiplication. Savage and Swamy [7] extended the method to show that $TS = \Omega(n^2)$ for binary integer multiplication and recently Savage [6] made further extensions, obtaining tradeoff results for a variety of problems including banded matrix multiplication.

In this paper, we present a spectrum of algorithms for n-bit integer multiplication that come close to meeting the lower bound of $TS = \Omega(n^2)$ [8] over a fairly wide range of space. The algorithms are modifications of the scheme due to Schonhage and Strassen [9], in which integer multiplication is done by treating blocks of bits in the integers as coefficients of two polynomials to be convolved, where the convolution of the sequences of coefficients is computed using back-to-back fast Fourier transforms and recursive invocations of the algorithm. We develop a graph representation of this algorithm, and prove, using techniques outlined in [2, 3], upper and lower bounds on the pebbling time of the graph associated with multiplying two n-bit numbers of the form $T = \theta(n^3/S^2 + n^2\log n \log\log n/S)$ when space $S \geq n^{1/2}\log n$ is used. These results imply a number of things, one of which is that time $T = O(n \log n \log\log n)$ can be achieved by such a convolution based algorithm only if space $S = \Omega(n)$. Also, there are algorithms using time $T = O(n^{1+\varepsilon})$ and space $S = O(n^{1-\varepsilon/2})$

($0 < \varepsilon < 1$ a real constant), i.e. $TS = O(n^{2+\varepsilon/2})$, which for small ε comes close to achieving the lower bound of $TS = \Omega(n^2)$ that must be satisfied by any integer multiplication algorithm. Our development of reduced space algorithms illustrates clearly the exchange between time and space indicated in the lower bounds of Savage and Swamy [8] by providing points close to the tradeoff curve other than the two extremes of and ($T = n \log n \log\log n$, $S = n$) (fastest known algorithm: [9]) and $T = n^2$, $S = \log n$ ('standard' algorithm: minimum space).

II A Graph Representation of High Speed Integer Multiplication

The traditional method for forming the product of two n-bit integers requires $O(n^2)$ bit operations. A simple way to speed this up is to partition each integer into two n/2-bit integers and use three (recursive) n/2-bit integer multiplications to form the product, resulting in time $O(n^{\log_2 3})$. This technique can be generalized by partitioning each integer into p blocks of q bits each, and treating the blocks as coefficients of a polynomial. The product of the two integers is then the convolution of the sequences of polynomial coefficients, which can be performed by evaluating the polynomials at a set of points (a forward transform), multiplying sample values (recursively), and interpolating (an inverse transform). When fast Fourier transform algorithms are used to perform the forward and inverse transforms, the method requires $O(n \log n \log\log n)$ bit operations [9]. The transform algorithms must be adapted to work in a finite domain using modular arithmetic, and in actuality, a wrapped convolution is computed to reduce the number of recursive invocations of the algorithm that must be performed. The algorithm can be summarized by the following set of steps, which is an adaptation similar to that given in [1;Ch.7].

Schonhage-Strassen Integer Multiplication Algorithm

Input: Two n-bit integers A and B where $n=2^k$ and k is even.

Output: The n+1-bit product of A and B modulo 2^n+1.

Preliminaries: Let $p = q = 2^{k/2}$. Partition A and B into p blocks of q bits each, i.e. $A = \sum_{0\leq i<p} a_i 2^{qi}$, $B = \sum_{0\leq i<p} b_i 2^{qi}$.

Step 1: Compute the Fourier transform, modulo $(2^{2q}+1)$ of $[a_0, \phi a_1, \ldots, \phi^{p-1} a_{p-1}]'$ and $[b_0, \phi b_1, \ldots, \phi^{p-1} a_{p-1}]'$ where $\phi = 2^{2q/p}$ (=4) and ϕ^2 is a principal pth root of unity.

Step 2: Compute the pairwise products modulo of $(2^{2q}+1)$ of the Fourier transforms computed in step 1, using recursive invocations of the algorithm.

Step 3: Compute the inverse Fourier transform modulo $(2^{2q}+1)$ of the vector of pairwise products produced in step 2. The result is $[c_0, \phi c_1, \ldots, \phi^{p-1} c_{p-1}]'$ modulo $(2^{2q}+1)$, where c_i is the ith term of the negative wrapped convolution of $[a_0, a_1, \ldots, a_{p-1}]'$ and $[b_o, b_1, \ldots b_{p-1}]'$. Thus, $c'_i=c_i$ modulo $(2^{2q}+1)$ can be computed by multiplying c_i by ϕ^{-i} modulo $(2^{2q}+1)$.

Step 4: Compute $c''_i=c_i$ modulo p by constructing two 0(p log p)-bit A′ and B′ from the coefficients a_i mod p and b_i mod p, and multiplying A′ and B′ using the simple partitioning method of complexity $0((p \log p)^{\log_2 3})$ (see [1;Ch.7] for the details of how to construct A′ and B′ and how to recover the c''_i s).

Step 5: Compute c_i modulo (2^n+1) using the formula $c_i= (2^{2q}+1)\ ((c''_i-c'_i)$ modulo p) + c'_i

Step 6: Compute AB modulo 2^n+1 = $\sum_{0 \leq i < p} c_i\, 2^{qi}$ modulo (2^n+1).

Steps 1 through 3 above form the heart of the Schonhage - Strassen integer multiplication algorithm, and when steps 1 and 3 are implemented using fast (0(n log n) bit operations) discrete transform methods, the result is T = 0(n log n loglog n). Later in the paper, the time and space complexity of steps 4 through 6 will be analyzed, and it will be seen that each can be performed in 0(n) bit operations using space $0(n^{1/2} \log n)$. Also, each operation of the p = $0(n^{1/2})$ component fast Fourier transforms used in steps 1 and 3 is a combination addition-multiplication (by a power of the principal root of unity) modulo $(2^{2q}+1)$, which can be done in $0(n^{1/2})$ bit operations using $0(n^{1/2})$ bits of storage space (note that the multiplication is simple a shift left operation). Thus, steps 1 and 3 each can be modeled by an FFT graph having p = $0(n^{1/2})$ inputs and outputs and $0(n^{1/2} \log n)$ nodes, where each node of the graph represents an operation of time and space complexity $0(n^{1/2})$. FFT graphs are explicitly defined in Savage and Swamy [7]; the definition is omitted here for lack of space.

Steps 1 and 2 produce the values of a product polynomial at powers of the principal root of unity, which is then interpolated by step 3. A careful analysis shows that the bit-reversal connection pattern is required to route the outputs of steps 1 and 2 to the inputs of the FFT graph model of step 3. The more interested reader is referred to Carlson [3] to see that this is indeed the case.

The only item that remains to be discussed before we define a graph model for steps 1 through 3 of the Schonhage-Strassen integer multiplication algorithm is the effect of step 2 on the model. Since step 2 is composed of recursive invocations of the algorithm, we can represent it as a recursive version of the defined graph. However, to make our later analysis of the graph simpler, we choose to represent each recursive invocation of the algorithm done in step 2 as a special node in the graph requiring certain values of time and space to pebble (much as each node of the FFTs requires time and space $O(n^{1/2})$). The specific time and space requirements of these nodes will be derived later in the paper.

Definition: The graph IM(n) associated with steps 1 through 3 of the Schonhage-Strassen n-bit integer multiplication algorithm is composed of three FFT graphs FFT(j), $0 \leq j \leq 2$, each having $p = n^{1/2}$ inputs and outputs, and special nodes $\{v_i \mid 0 \leq i < p\}$. For $0 \leq i < p$, output i of FFT(0) and FFT(1) is connected to node v_i, and node v_i is connected to input bitrev(i) of FFT(2). Nodes of the FFT graphs represent addition-multiplication modulo $(2^{2q}+1)$ operations performed in the algorithm, and the nodes $\{v_0, \ldots, v_{p-1}\}$ represent recursive invocations of the algorithm. Figure 1 illustrates the structure of IM(n).

III. Upper and Lower Bounds on Time and Space

In this section, we consider the time required to pebble the graph IM(n) using certain amounts of space. We develop efficient pebbling strategies for the graph, and argue that they are optimal over all FFT-based integer multiplication algorithms by deriving lower bounds on the graph's simultaneous time and space requirements.

As seen in Section 2, IM(n) is very similar to a back-to-back FFT graph having a bit-reversal connection pattern between the individual FFT graphs. The time-space tradeoff characteristics of these graphs were studied in Carlson [2,3], whose results are summarized here in the following two lemmas.

Lemma 1 (Carlson [2]): For a k input back-to-back FFT graph having the bit-reversal connection pattern, when S pebbles are used, pebbling time grows as $T = \Omega(k^3/S^2 + k^2\log k/S)$.

Lemma 2 (Carlson [3]): For a k input back-to-back FFT graph having the bit-reversal connection pattern, there are pebbling strategies that use S pebbles and time $T = O(k^3/S^2 + k^2\log k/S)$.

Combining Lemmas 1 and 2 above yields matching upper and lower bounds on pebbling time, and thus we can conclude that there are pebbling strategies for back-to-back FFT graphs that are optimal with respect to time and space. Here, we apply similar techniques to derive bounds on the time required to pebble the graph IM(n) using a certain amount of space. Our main result is contained in the following theorem.

Theorem 1: For space (number of bits) S between $\Omega(n^{1/2}\log n)$ and $O(n)$, n-bit integer multiplication can be performed using time $T = O(n^3/S^2 + n^2\log n \log\log n/S)$. Furthermore, the upper bound on time is optimal over all convolution based methods.

Proof: We split the proof of the theorem into the following three propositions. Proposition 1 derives upper and lower bounds on the pebbling time of the graph IM(n), Proposition 2 deals with the time and space complexities of the individual operations modeled by nodes of the graph, and Proposition 3 analyzes the time and space complexities of steps 4 through 6 of the Schonhage-Strassen algorithm described in Section 2.

Proposition 1: Let α be the amount of time required to pebble each node of the FFT graphs and let β be the amount of time required to pebble each node $\{v_i\}$ in IM(N). When $S = \Omega(\log n)$ pebbles are used to pebble the graph, each representing $O(n^{1/2})$ bits, the time required grows as $T = \theta(\alpha p^3/S^2 + \alpha p^2 \log p/S + \beta p^2/S)$ where $p = n^{1/2}$.

Proof: Using ideas from Savage and Swamy [7] and Carlson [2], we first derive the lower bound. The graph IM(n) is easily seen to contain a back-to-back FFT graph on p inputs and outputs. If each node had unit cost to pebble, it would follow that $T = \Omega(p^3/S^2 + p^2\log p/S)$. To account for the pebbling costs of nodes in IM(n), we consider the ideas leading to the above lower bound. From Savage and Swamy [7], pebbling any set of O(S) outputs of FFT(2) (and thus of IM(n)) requires the pebbling of $\Omega(p)$ inputs of FFT(2) when S pebbles are available. The inputs of FFT(2) are direct successors of the nodes $\{v_i\}$, and thus $\Omega(p)$ nodes $\{v_i\}$ also must be pebbled. To analyze the amount of work done on FFT(0) and FFT(1), we can either split the $\Omega(p)$ outputs of FFT(0) and FFT(1) that must be pebbled from above into subsets of size $O(S)$, applying the same argument as before, or argue as in Carlson [2] that $\Omega(p \log p)$ moves are required at a minimum to complete this task. Combining these ideas with the time complexity of pebbling each node yields:

$$\begin{aligned} T &= \Omega((\alpha p + \beta p + \alpha p \log p + \alpha p(p/S))p/S) \\ &= \Omega(\alpha p^3/S^2 + \alpha p^2 \log p/S + \beta p^2/S) \end{aligned}$$

To derive pebbling strategies for IM(n) that meet the above lower bound, we first note that the nodes $\{v_i\}$ are connected to inputs of FFT(2) by the bit reversal connection pattern. As shown in Carlson [3], this implies that the pebbling strategy for an individual FFT graph due to Savage and Swamy [7] can be applied to the graphs FFT(0) and FFT(1) to pebble their outputs in the order required by the pebbling strategy used on FFT(2). The pebbling strategy for an individual p-input FFT (with unit cost moves) requires time $T = O((S \log S + p)(p/S))$ when $\theta(S)$ pebbles are used. Taking into account the time complexity to pebble the nodes of IM(n), we obtain a pebbling strategy using $S = \Omega(\log n)$ pebbles (note that the minimum space requirement of IM(n) is log n pebbles) and time

$$\begin{aligned} T &= O([\alpha(S \log S + p) + \beta p + \alpha(S \log S + p)p/S]p/S) \\ &= O(\alpha p^3/S^2 + \alpha p^2 \log p/S + \beta p^2/S) \end{aligned}$$

QED

Proposition 2: For space (number of bits) S between $\Omega(n^{1/2}\log n)$ and $O(n)$, steps 1 through 3 of the Schonhage-Strassen integer multiplication algorithm require time $T = \theta(n^3/S^2 + n^2 \log n \log\log n/S)$.

Proof: In Proposition 1, we saw that time $T = \theta(\alpha p^3/S^2 + \alpha p^2 \log p/S + \beta p^2/S)$ is required to perform steps 1 through 3 of the Schonhage-Strassen integer multiplication algorithm when S pebbles, each representing $\theta(n^{1/2})$ bits of storage, are used to pebble the associated graph IM(n). Here, α and β are the time complexities of pebbling the two types of nodes contained in IM(n). It is easily seen that $\alpha = \theta(n^{1/2})$, since the combination addition-multiplication operations performed in the fast Fourier transforms of steps 1 and 3 can be performed in time proportional to the number of bits in each component of the input vector (see [1;Ch.7] for a detailed analysis of this operation). Also, it is realistic to associate pebbles with groups of $\theta(n^{1/2})$ bits of storage, since the results of intermediate computations in the FFTs require this much storage to hold in their entirety, and must be totally recomputed if any portion of their $\theta(n^{1/2})$ bits are released from temporary storage.

The nodes $\{v_i\}$ in IM(n) represent recursive invocations of the algorithm, the problem size being $2q = \theta(n^{1/2})$. thus, when our fast integer multiplication algorithm is applied to do this recursive computation, time and space on the order of $\theta(n^{1/2}\log n \log\log n)$ and $\theta(n^{1/2})$ are required, respectively. Note that since space $\Omega(n^{1/2})$ is available to pebble individual nodes of the FFT graphs of steps 1 and 3, it is also available for pebbling the nodes $\{v_i\}$ so that restricted space implementations for the recursive invocations of the algorithm are not needed; we simply perform them as fast as possible. Therefore, $\beta = \theta(n^{1/2}\log n \log\log n)$ and since the actual space used in the algorithm is $\theta((\#\text{pebbles})\, n^{1/2})$, we obtain

$$T = \theta(\alpha p^3/(\#\text{pebbles})^2 + \alpha p^2 \log p/(\#\text{pebbles}) + \beta p^2/(\#\text{pebbles}))$$
$$\theta(n^3/S^2 + n^2 \log n \log\log n/S)$$

where S is the number of bits of storage used. QED

Proposition 3: Steps 4 through 6 of the Schonhage-Strassen integer multiplication algorithm each can be performed in time $O(n)$ using space (number of bits) $O(n^{1/2}\log n)$.

Proof: The claim is obvious for steps 5 and 6. The time complexity of step 4 is $O((p \log p)^{\log_2 3})$ and since $p = n^{1/2}$, this is $O(n)$. The algorithm of step 4 is implementable using space $O(p \log p) = O(n^{1/2}\log n)$.

QED

The results of Theorem 1 imply the existence of algorithms for integer multiplication that are efficient with respect to time for reduced values of space. In fact, since it was shown in Proposition 2 that any scheme based on back-to-back FFTs and recursive invocations must use time $T = \Omega(n^3/S^2 + n^2\log n \log\log n/S)$ when S bits of storage are available, the algorithms derived are optimal over the class of FFT-based implementations. For certain values of space, the upper bound of $T = O(n^3/S^2 + n^2\log n \log\log n/S)$ also comes close to meeting the lower bound of $T = \Omega(n^2/S)$ that must be satisfied by arbitrary straight-line algorithms for integer multiplication. If $S = O(n^{1-\varepsilon})$ then we have $T = O(n^{1+2\varepsilon})$ for our algorithms, whereas the general lower bound is $T = O(n^{1+\varepsilon})$, which is a close match for small ε. However, just as it is still an open question whether the time bound of $T = O(n \log n \log\log n)$ for integer multiplication can be improved upon, there is still a gap between our bounds of $T = O(n^3/S^2 + n^2 \log n \log\log n/S)$ and $T = \Omega(n^2/S)$. One other fact worth noting about our results is that they imply that the space requirements of the $O(n \log n \log\log n)$ integer multiplication algorithm cannot be reduced without a corresponding increase in time. This follows since $T = \Omega(n^3/S^2 + n^2 \log n \log\log n/S)$ (for convolution based methods), so that if $S = o(n)$, $T = \omega(n \log n \log\log n)$.

IV Conclusions

We have described a class of time and space efficient algorithms that implement n-bit integer multiplication. When $S = \Omega(n^{1/2} \log n)$ bits of storage are available, these algorithms require time $T = O(n^3/S^2 + n^2 \log n \log\log n/S)$, which is optimal over all FFT-based implementations and comes close to meeting the general lower bound of $T = \Omega(n^2/S)$ [8]. The results also imply that the space requirements of the $T = O(n \log n \log\log n)$ algorithm due to Schonhage and Strassen [9] cannot be reduced without affecting the time performance of the algorithm.

References

1. A. Aho, J. Hopcroft, and J. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, MA (1974).
2. D. A. Carlson, 'Time-Space Tradeoffs On Back-to-Back FFT Algorithms,' IEEE Trans. Comp., Vol. C-32, No. 6, pp. 585-589, (June 1983).
3. D. A. Carlson, 'Time-Space Tradeoffs for Back-to-Back FFTs: The Effect of the Connection Pattern,' Proc. 20th Allerton Conference on Communication, Control, and Computing, pp. 374-383 (Oct. 1982).
4. D. Y. Grigoryev, 'An Application of Separability and Independence Notions for Proving Lower Bounds on Circuit Complexity,' Notes of Scientific Seminars, Steklov Math. Inst., Leningrad, Vol. 60, (1976).
5. J. JaJa, 'Time-Space Tradeoffs for Some Algebraic Problems,' JACM, Vol. 30, No. 3, pp. 657-667, (July 1983).
6. J. E. Savage, 'Space-Time Tradeoffs for the Solution of Banded Linear Equations and other Problems,' Tech. Rep. CS-77, Dept. Comp. Sci., Brown University, Providence RI (July 1981).
7. J. E. Savage and S. Swamy, 'Space-Time Tradeoffs on the FFT Algorithm,' IEEE Trans. Info. Th., Vol. IT-24, No. 5, pp. 563-568 (Sept. 1978).
8. J. E. Savage and S. Swamy, 'Space-Time Tradeoffs for Oblivious Sorting and Integer Multiplication,' Proc. 5th Int. Colloq. on Aut., Lang., and Prog., Lecture Notes in Comp. Sci., Vol. 71, (ed. H. A. Maurer) Springer-Verlag, NY, pp. 498-504 (1979).
9. A. Schonhage and V. Strassen, 'Schnelle Multiplication Grosser Zallen,' Computing, Vol. 7, pp. 281-292 (1971).
10. M. Tompa, 'Time-Space Tradeoffs for Computing Functions, Using Connectivity Properties of their Circuits,' JCSS, Vol. 20, pp. 118-132 (April 1980).

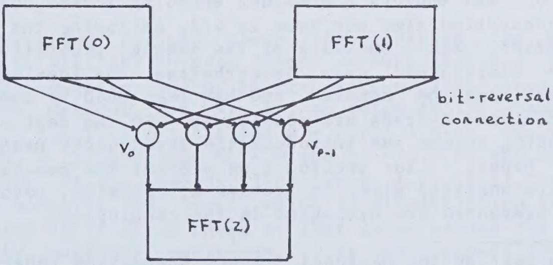


Figure 1: The graph IM(n)

GENERATION OF BINARY TREES OF BOUNDED HEIGHT

C. C. LEE[1], D. T. LEE[1*] AND C. K. WONG[2]

1. Department of Electrical Engineering and Computer Science
Northwestern University, Evanston, IL 60201

2. IBM T. J. Watson Research Center, P. O. Box 218
Yorktown Heights, NY 10598

Abstract

In this paper we consider the problem of generating binary trees with n keys (hence n + 1 leaves) whose height is no greater than a constant h. Using a level-by-level encoding scheme we are able to enumerate all possible trees in a certain predetermined order. It is also proved that the average generation time per tree is constant.

1. Introduction

The problem of generating, ranking and unranking binary trees, k-ary trees and other combinatorial objects has been studied extensively in the past few years[1-3,6,7,9-18]. In this paper we consider a new class of binary trees with n keys, (and hence n+1 leaves) whose height is bounded above by a constant h. The height of a binary tree T with n keys, denoted $\text{height}(T_n)$, is defined to be the number of edges of a longest path from the root to a leaf and satisfies the inequality $\lceil \log_2(n+1) \rceil \le \text{height}(T_n) \le n$. When the bound h is nontrivial, i.e., $\lceil \log_2(n+1) \rceil \le h \le n$, there is no closed form expression known for the number of binary trees with n keys whose height is bounded above by h. However, an approximate upper bound has been obtained[4]. We shall study the problem of enumerating all possible trees in certain predetermined order, and prove that the average generation time per tree is a constant.

In order to enumerate the trees we need to have a proper representation of a tree. Given a tree as shown in Fig. 1, in which the leaves are drawn as squares and (internal) nodes are drawn as circles, one way to encode the tree is by **levels** as 11101110011000000, where 1's represent the nodes and 0's the leaves. Obviously, this encoding will require 2n + 1 bits and has exactly n 1's and n + 1 0's. Note that not every bit string represents a tree. The string that represents a tree has to satisfy the so-called **prefix property** that in any prefix of the string the number of 1's is no less than the numbers of 0's. (Here we have ignored the last bit which is always 0.) From the prefix property it follows that the total number of trees with n nodes is the Catalan number $B_n = \frac{1}{n+1}\binom{2n}{n}$ [6,8]. For the unconstrained case where no bound on the height is given, this property is used by Zaks[16], who employs a preorder encoding scheme and has shown that the average generation time per tree is 4/3, excluding the time needed for output. However, Zak's analysis of the average generation time does not apply to the constrained case. Nevertheless, the idea of level-by-level encoding scheme can be adapted, and has been used by Gupta et. al.[2,3] when 2-3 trees and B-trees are considered. In the next section we shall give our encoding scheme and introduce the terminology used throughout the rest of the paper. In section 3 we present the generation algorithm, followed by its analysis given in section 4. Finally, possible extensions to the result presented are discussed in the conclusion.

---

*Supported in part by the National Science Foundation under the Grants MCS 8202359 and ECS 8121741.

2. Encoding scheme and ordering

We shall deal only with binary trees T_n of fixed height, since binary trees of bounded height h can be generated by decreasing heights, i.e., binary trees of height h, followed by those of height h - 1, h - 2, ... until $\lceil \log_2(n+1) \rceil$. Let us assume that the height is h. Instead of using bit string to represent a binary tree as described above, one can use 'position array', i.e., $p_1, p_2, \ldots, p_n$, where p_i is the position of the i^{th} 1 in the bit string. For example, the tree given in Fig. 1 is encoded as 1,2,3,5,6,7,10,11. Since we encode the tree level by level, if N(k) denotes the number of nodes at level k, $1 \leq k \leq h$, the number of nodes at level k+1 is $N(k+1) \leq 2N(k)$, for $1 \leq k < h$. Thus, the above 'position array' encoding can be modified so that the positions of 1's within each level are recorded, rather than the positions of 1's with respect to the entire bit string. With this modification the encoding of the tree given in Fig. 1 becomes 1;1,2;2,3,4;3,4, where ;'s are used to delimit the level boundaries. Note that the position number p_i at level k must satisfy $1 \leq p_i \leq N(k)$. To help distinguish the level boundaries we use arrays N(k), $1 \leq k \leq h$, which is to be included as part of the encoding. When $N(k) = 2N(k-1)$, we say that the kth level is **filled**, and **unfilled** otherwise. For a filled level the positions of the nodes are completely determined by their parents and hence the corresponding position array can be omitted. Since the root level always contains a 1, it will also be omitted. For instance, the main sequence for the tree shown in Fig. 2 is M = 1,2,4;3,5;4 and the associated node distribution sequence is N = 1,2,3,2,4,1.

To record the fact that some filled levels are omitted, we use array A(k), where A(1) denotes the **bottommost** level number of an unfilled level, A(2), that of the second unfilled level from the bottom, etc. In the example above we have the A(k) sequence A = 1,3,4. Thus, the main sequence M is a concatenation of $|A|$ position vectors. Given the sequences M, N and A, we can uniquely determine a tree, if appropriate conditions are met, and vice versa. (Note the two sequences M and N are sufficient to have this property.) The sequences M, N and A for which a tree can be reconstructed are referred to as feasible sequences and only feasible sequences will be considered in the sequel.

The ordering of the trees, specified by sequences N,M, is defined as follows. Let T1 and T2 denote two trees of n nodes and of height h. (The subscript n for T is dropped without any confusion.) T1 < T2 if and only if (i) N1 is lexicographically less than N2, or (ii) N1 = N2 and M1 is lexicographically less than M2, where Ni and Mi denote, respectively, the N and M sequences of tree Ti, i = 1,2. We say that T2 is the successor of T1, denoted by T2 = SUCC(T1), if T1 < T2 and there is no other T such that T1 < T < T2.

3. Generation algorithm

With the ordering defined above, to generate the next tree T2, SUCC(T1), of T1 we will try to keep intact as many levels of M1 as possible by changing the position vector of the last unfilled level, if M1 is not lexicographically maximum for N1. Let us refer to the example of Fig. 2, where N = 1,2,3,2,4,1, A = 1,3,4 and M = 1,2,4;3,5;4. The successor tree has M'= SUCC(M) = 1,2,4;3,5;5, N' = N and A' = A. To obtain M' from M we shall first look at the last unfilled level, i.e., A(1), and then obtain the number of nodes at that level, which is N(h-A(1)+1)=N(6)=1. The next thing to examine is if these nodes at that level assume the rightmost (or lexicographically maximum) positions. In our example the only node at level A(1) is in position 4 and is not the rightmost, which is 2N(5)=8, we increment the position by 1, thereby shifting the node one position to the right and obtaining the desired tree. If M were 1,2,4;3,5;8, then we would need to consider the next unfilled level from the bottom, i.e., $A(2)=3$. In this case, from $N(h-A(2)+1)=N(4)=2$ we know this level has two nodes positioned at 3 and 5. Since these two nodes are not at the lexicographically maximum positions, we compute their successor positions, which are 3 and 6. As soon as the position array is changed at a level, which is not the bottommost one, all the nodes at the succeeding levels must be positioned at the lexicographically minimum positions, i.e., left justified. Therefore, the successor of M = 1,2,4;3,5;8 is M' = 1,2,4;3,$\underline{6}$;$\underline{1}$, where the underlined positions indicate those that are different from the previous M.

As can be seen, often times we need to scan the array A to find the level in which the nodes are not at the lexicographically maximum positions, and when such a level is found, we need to know further **which** node at this level can be moved one position to the right. To speed up the location of such a node we keep an additional array D. D(k) is associated with A(k) and is 0 if all the nodes at level A(k) are at the rightmost positions, and is the position at which the rightmost "moveable" node is, otherwise. (Note that the indexes for arrays A and D are associated with levels from bottom up.) In case the main sequence M for which the successor tree M' is sought is lexicographically maximum, we need from the definition of our ordering on trees to generate the successor N' of the sequence N, if N' exists, (or if N is not lexicographically maximum) and then compute the lexicographically minimum main sequence associated with N'. We shall deal with this case later in the section. The following is an algorithm for generating the successor tree of a given tree, whose main sequence M is not lexicographically maximum, by changing only the portion of M that needs to be modified.

Algorithm NXTREE(M,N,A,D) Recall that we have a main sequence M, associated with an array N, along with two auxiliary arrays A and D. Assume that M is not lexicographically maximum. This algorithm returns the successor sequence of M.

```
1. Find the first nonzero D(k). Let $D(k) \neq 0$, $k \leq h$ and $D(i) = 0$, $0 < i < k$
2. Compute the position p in the main sequence and increment it by 1. Let
   the length of the encoding be $L = \sum_{j=1}^{h} N(A(j))$.
   $p = L - \sum_{j=1}^{K} N(A(j)) + D(j) - 1$; $M(p) = M(p) + 1$.
3. Place all the nodes of levels A(i), $0 < i < k$, at the leftmost positions
   and update D(i)'s accordingly. Note that the same remark applies also
   to the nodes at level A(k) that appear after node p.
   Let STOP be the position of the rightmost node at the same level as p in
   the main sequence.
   STOP = p + N(A(k)) - D(k);
   pl = p + 1;
   if $D(k) \neq N(A(K))$ {Current node is not rightmost at that level.}
3a. then if M(pl) - M(p) = 1
         then D(k) = D(k) - 1 {"Gap" moves to immediate left.}
3b.      else begin D(k) = N(A(k))  D(k) is set to the rightmost position.
                    while $pl \leq$ STOP do begin M(pl) = M(p ) + 1
                                                  p = pl
                                                  pl = p + 1
                                            end
                    Nodes after the current one are 'left-justified'.
              end
   else if M(p) = 2N(A(k)-1)  Current node becomes the rightmost.
        then D(k) = D(k) -1
         else D(k) remains unchanged.
```

```
    p = STOP + 1
3c. For i = k-1 step -1 to 1 do
        Move the nodes at subsequent levels to the leftmost positions.
        Set D(i) at the rightmost position at each (unfilled) level.
      begin for j = 1 to N(A(i)) do
              begin M(p) = j
                    p = p + 1
              end
          D(i) = N(A(i))
      end
```

Let us now consider the case where the main sequence M for a given N is lexicographically maximum. If N itself is not lexicographically maximum, then we should find its successor N' in the ordering and generate a lexicographically minimum main sequence M'. When N is lexicographically maximum, then we have reached the lexicographically maximum tree of height h. The trees of height h-1 will then be generated if they exist.

Given an N, its associated lexicographically minimum main sequence $M = m(1)\|\|m(2)\|\|\ldots\|\|m(h)$, a concatenation of h subsequences m(i), where m(i) is a sequence of consecutive integers from 1 to N(i), i=1,2,...,h. However, since we are using simplified encoding, we will exclude from M the sequences that correspond to filled levels. The arrays A and D can also be determined easily, i.e., array A records from the bottom the level numbers of those **unfilled** levels and each entry in array D is set to the position of the rightmost node at each unfilled level. (Recall that a level N(i) is unfilled if $N(i) < 2N(i-1)$, i=2,...,h.) For example, let N = 1,2,3,2,4,1. The lexicographically minimum main sequence is 1,2,3;1,2;1, and A = 1,3,4; D = 1,2,3.

Before we describe the generation of the successor N' of a given N, let us consider the problem of getting the lexicographically minimum N sequence, given n and the height k. Assume that $\lceil \log_2(n+1) \rceil \leq k \leq n$. Informally, consider a matrix which has k columns and unbounded number of rows. The problem we have is equivalent to assigning, say n 0's, to the matrix column by column as follows. Initially, each column receives exactly one zero, as shown in Fig. 3a, in which we have n = 24, k = 6, and the first row has been filled with zeros. Subsequent assignments of zeros must follow the rule: always assign a zero to the rightmost column, wherever possible, subject to the condition that the number of zeros at the current column receiving the zero does not exceed twice the number of zeros in the previous column. Fig. 3b shows the final assignment of 24 zeros and Fig. 3c shows the order in which the zeros are assigned after the initial assignment. After all zeros have been assigned, the sequence N that we look for will be the number of zeros in each column. For the example shown in Fig. 3, the sequence N is 1,1,2,3,6,11.

We summarize the previous discussion below, followed by a formal description of the algorithm.

1. Compute the largest r such that $(n-k) \geq 2^r - r - 1$, and obtain an initial sequence $N = 1,1,\ldots,1,2^0,2^1,\ldots,2^{r-1}$, where the number of 1's is k - r. For convenience, let us denote N by $1^{(k-r)},2^0,2^1,\ldots,2^{r-1}$.

   This step basically computes the maximum number of columns in which we have an assignment that assigns 1, 2, 4, ..., 2^{r-1} zeros to the last r columns respectively. In our example of Fig. 3, r is 4 and the pattern of zeros is enclosed in the region as shown in Fig. 4.

2. Let $n' = (n-k) - (2^r-r-1)$.

   n' is the remaining number of zeros after the assignment of $k+2^r-r-1$

zeros as shown in Fig. 4.

3. If $n' \leq r$ then N is obtained by adding $0^{(k-r)},1^{(n')},0^{(r-n')}$ to the old N, component by component, and becomes $1^{(k-r)},2^{0}+1,2^{1}+1,\ldots,2^{n'}+1,2^{n'+1},\ldots,2^{r-1}$. We are done.
4. We now have an instance of the original problem, i.e., given n' and r, find the lexicographically minimum sequence N'.

   Compute the largest $r'(< r)$ such that $(n'-r) \geq 2^{r'} - r' - 1$ and obtain $N' = 1^{(r-r')},2^{0},2^{1},\ldots,2^{r'-1}$. This N' is to be 'added' (right-justified)(1) to the old N, and the process repeats.

Algorithm COMPUTE-N(n,k)(2) This is a recursive algorithm which returns the lexicographically minimum N when given n and k as input.

```
1. If $n \leq k$ then return $1^{(n)},0^{(k-n)}$
2. Compute the largest r such that $(n-k) \geq 2^{r} - r - 1$.
3. Let $n' = (n-k) - (2^{r}-r-1)$.
4. return $(1^{(k-r)},2^{0},2^{1},\ldots,2^{r-1}$ + COMPUTE-N(n',r)).
   The + operation is a right-justified addition.
```

In our example in which n=24 and k=6, we will have generated three sequences when we call COMPUTE-N(24,6) and they are 1,1,1,2,4,8 and 1,1,1,2 and 1,1. When these three sequences are added right-justified, we have the answer 1,1,2,3,6,11.

We are now ready to give the algorithm for computing the successor N' of a given N. As before, we shall keep intact as much as possible the left portion of the array N. We therefore scan the array from right to left and stop at the first position encountered that can be incremented by 1. Since each number N(i) denotes the number of nodes at level i, incrementation of N(i) means deletion of a node from the lower levels, i.e., levels i+1, i+2, etc. Therefore, the last position N(k) is not incrementable. In general, a position N(i) is incrementable if $N(i) < 2N(i-1)$ and m', the total number of nodes at levels i+1, ..., k, is greater than k - i. Suppose that N(i) is incrementable. Then we have $m = m' - 1$ nodes to be distributed to k-i lower levels with each level receiving at least one node. Since the subsequence N'(i+1),N'(i+2),...,N'(k) should be lexicographically minimum, the problem is similar to that of computing the lexicographically minimum sequence given (m,k-i), with the exception that the first position, N'(i+1), is not necessarily 1 as in the previous case. We give the algorithm below.

---

(1) We pad sufficient number of zeros at the beginning of the shorter sequence so that the lengths of the two sequences to be added are the same and we carry out the addition element by element.

(2) The recursive algorithm can be implemented nonrecursively just as the algorithm NEXT-N below so that the time involved is proportional to k.

ALGORITHM NEXT-N(n,k) {This algorithm returns a new sequence N' which is the successor of N for which the parameters are n and k, i.e., the number of nodes and the height, respectively.}

```
1. m = N(k)
2. i = k-1
   {i denotes the first incrementable position from the right.}
3. while N(i) = 2N(i-1) or $m \le k - i$ do
        begin m = m + N(i)
              i = i - 1
              if i = 0 then return 0
              {The sequence N is lexicographically maximum.}
        end
4. m = m - 1
{m is the number of nodes to be distributed to levels i+1,i+2,...,k.}
5. for j = 1 to i-1 do N'(j) = N(j) {Leftmost i-1 levels remain the same.}
6. N'(i) = N(i) + 1 {Increment the number of nodes at the $i^{th}$ level by 1.}
7. {Start distribution of nodes to levels i+1 through k.}
   for j = i to k-1 do
       begin Compute the largest r such that $m > r(2^{(k-j)} - 1)$
             N'(j) = r + 1
             m = m - N'(j)
       end
8. N'(k) = m
```

Generation algorithm of trees with n nodes and of bounded height h. We shall generate these n-node trees in decreasing height, trees of heights h, h-1, ..., $\lceil \log_2(n+1) \rceil$. For trees of the same heights, they are generated in lexicographically ascending order as defined in Section 2.

```
1. For k = h step -1 to $\lceil \log_2(n+1) \rceil$ do
2.  begin N = COMPUTE-N(n,k)
3.      Compute the lexicographically minimum main sequence M, and its
        associated A and D.
4.      Output (N,M,A,D)
5.      While M is not lexicographically maximum do
             begin M = NXTREE(M,N,A,D)
                   Output (M,D)  Arrays N and A remain unchanged.

             end
6.      if N is not lexicographically maximum
        then begin N = NEXT-N(n,k)
                   goto step 3.
             end
  end
```

4. Analysis

We first give the analysis of the algorithm NXTREE step by step.

Step 1. Searching for the first nonzero D(k).

$\text{Prob}\{ D(i) = 0 \} = b_i = \dfrac{1}{\binom{2n_{i+1}}{n_i}} \leq 1/2$, where n_i is the number of nodes at level i from the bottom.

$\text{Prob}\{ \text{first } j \text{ D's} = 0 \} \leq (1/2)^j$.

Thus, the average search time is

$$S = \sum_{j \text{ over unfilled levels}} j * \text{Prob}\{ \text{first } j \text{ D's} = 0 \}$$

$$\leq 1/2 + 2*(1/2)^2 + 3*(1/2)^3 + \ldots + (k-1)*(1/2)^{(k-1)} < 2.$$

Step 2. Constant time operation.

Step 3. Generating time of the main sequence for next tree (Steps 3b and c)

Consider an unfilled level j from the bottom.

Prob D(j) is at the i^{th} rightmost position $= \text{Prob}\{D(j) = n_j - (i-1)\}$

$$= p_1 p_2 \ldots p_{i-1}(1-p_i), \text{ where } p_i = \frac{n_j - (i-1)}{2n_{j+1} - (i-1)}, \ i = 1,2\ldots,n_j$$

Assume that D(j) is the first nonzero number and that the i^{th} rightmost node at level j from the bottom is just incremented. Note that if the test in line 3a is true, then line 3b is skipped. The probability of this occurring is $p_1 p_2 \ldots p_i (1-p_{i+1})$, since this happens only when D(j) becomes $n_j - i$ and $\text{Prob}\{D_j = n_j - i\}$ is $p_1 p_2 \ldots p_i (1-p_{i+1})$. Therefore, the expected time for generating the main sequence at level k is

$$E_j = \sum_{i=1}^{n_j} [i \, p_1 p_2 \ldots p_{i-1}(1-p_i) - (i-1) p_1 p_2 \ldots p_i (1-p_{i+1})] + n_j p_1 p_2 \ldots p_{n_j}$$

with $p_{n_{j+1}} = 0$.

$$E_j = 1 + p_1 - p_1 p_2 \ldots p_{n_j}$$

$$= 1 + \frac{n_j}{2n_{j+1}} - \frac{1}{\binom{2n_{j+1}}{n_j}} = 1 + c_j - b_j$$

Since level j will be involved in the generating process if and only if the D's below level j are all zeros, i.e., D(i) = 0 for all 0 < i < j, the total expected time is

$$E = \sum_k E_k * \text{Prob}\{ D(i) = 0, \ 0 < i < k\} = \sum_k E_k \left(\pi_{0<i<k} b_i\right)$$

Note that $c_j = \dfrac{n_j}{2n_{j+1}} \leq \dfrac{2n_{j+1}-1}{2n_{j+1}} = 1 - \dfrac{1}{2n_{j+1}} = 1 - e_j$, and

$b_j = \dfrac{1}{\binom{2n_{j+1}}{n_j}} \leq \dfrac{1}{2n_{j+1}} = e_j \ (\leq 1/2)$.

We have

$$
\begin{aligned}
E &= 1+c_1+b_1c_2+b_1b_2c_3+\ldots+b_1b_2\ldots b_{k-1}c_k-b_1b_2\ldots b_k \\
&\leq 1+(1-e_1)+e_1(1-e_2)+e_1e_2(1-e_3)+\ldots+e_1e_2\ldots e_{k-1}(1-e_k)-e_1\ldots e_k \\
&= 2 - e_1\ldots e_k < 2.
\end{aligned}
$$

As for the expected generation time for D(j)'s, we have $F = 1*1/2 + 2*1/4 + 3*1/8 + \ldots + j*1/2^j < 2$.

We therefore conclude that the total expected time for generating the next main sequence is at most $S + E + F = 6$.

Now let us analyze the expected time for the generation of the next node distribution sequence, given an N, where $N(k) = n_k$ denotes the number of nodes at level k, k= 1, 2, ..., h.

We state without proof the following:

Lemma 1: Let $T(n_1,n_2,\ldots,n_h)$ denote the number of trees associated with the node distribution sequence N(k), k= 1, 2, ..., h.

Then $T(n_1,n_2,\ldots,n_h) = \binom{2n_1}{n_2}\binom{2n_2}{n_3}\cdots\binom{2n_{h-1}}{n_h}$

Lemma 2: If $N(h) < 2N(h-1)$, i.e., the bottommost level is unfilled, then $T(n_1,n_2,\ldots,n_h) > n/2$.

Lemma 3: Let S_1 and S_2 denote the sets of node distribution sequences with $N(h)=2N(h-1)$ and $N(h)<2N(h-1)$, respectively. If $n \neq 2^h-1$, then $\P S_1 \P \leq \P S_2 \P$.

Corollary: Let $T(n_1,n_2,\ldots,n_h)$ denote the number of trees with node distribution sequence $N(i)=n_i$, $i=1,2,\ldots,h$. Then the average number of trees per node distribution $E[T(n_1,n_2,\ldots,n_h)] > n/4$, where $n = \Sigma_i N(i)$.

With this corollary in mind, we can see that on the average each time when step 3 of the generation algorithm is executed, the while loop at step 5 will be executed at least n/4 times. Since the algorithms COMPUTE-N and NEXT-N run in time proportional to the length of the sequence, i.e., the current height of the tree being generated, each main sequence generated is charged, on the average, no more than a constant (4) units of time. Thus, we have our main theorem as follows.

Theorem: The binary trees with n nodes of bounded height h can be enumerated in constant time per sequence generated.

5. Conclusion

We have studied a new class of binary trees with n keys and of bounded height h, and shown that the trees within this class can be generated in order in constant average time per tree. A natural question that arises is how we generate a random tree within this class and how much time it takes. If an exact number of the trees is known, then selection of a random tree can be obtained via a random number generator which generates a number between 1 and ¶T(n,h)¶, where ¶T(n,h)¶ denotes the total number of trees of this class, and the tree corresponding to the number (the rank) is found. Companion to the question is that of determining the rank of a given tree. Techniques used in [2,3], for example, can be applied to the ranking and unranking problems without knowing the exact number of trees of this class. Whether or not one can generate this class of trees and perform ranking and unranking operations in a more efficient way is an open problem.

Generalization of the technique to other combinatorial objects such as t-ary trees, 2-3 trees, etc. is currently under investigation.

References

1. T. Beyer and S. M. Hedetniemi, "Constant time generation of rooted trees," SIAM J. Comput., 9 (1980), 706-712.

2. U. I. Gupta, D. T. Lee and C. K. Wong, "Ranking and unranking of 2-3 trees," SIAM J. Comput., 11 (1982), 582-590.

3. U. I. Gupta, D. T. Lee and C. K. Wong, "Ranking and unranking of B-trees," J. Algorithms, 4 (1983), 11-60.

4. P. Flajolet and A. Odlyzko, "The average height of binary trees and other simple trees," J. Comput. and System Sci., 25 (1982), 171-213.

5. G. D. Knott, "A numbering system for binary trees," Comm. ACM, 20 (1977), 113-115.

6. A. Nijenhuis and H. S. Wilf, Combinatorial Algorithms, Academic Press, London/New York, 1978.

7. A. Proskurowski, "On the generation of binary trees," J. ACM, 27 (1980), 1-2.

8. E. M. Reingold, J. Nievergelt, and N. Deo, Combinatorial algorithms - Theory and Practice, Prentice-Hall, Englewood Cliffs, New Jersey, 1977.

9. D. Rotem and Y. L. Varol,"Generation of binary trees from ballot sequences," J. ACM 25 (1978), 396-404.

10. F. Ruskey, "Generating t-ary trees lexicographically," SIAM J. Comput., 7 (1978), 424-439.

11. F. Ruskey and T. C. Hu, "Generating binary trees lexicographically," SIAM J. Comput., 6 (1977), 745-758.

12. M. Solomon and R. A. Finkel, "A note on enumerating binary trees," J. ACM 27 (1980), 3-5.

13. A. E. Trojanowski, "Ranking and listing algorithms for k-ary trees," SIAM J. Comput., 7 (1978), 492-509.

14. H. S. Wilf, "A unified setting for sequencing, ranking and selection algorithms for combinatorial objects," Advances in Math., 24,3 (1977), 281-291.

15. S. G. Williamson, "On the ordering, ranking and random generation of basic combinatorial sets," in Combinatoire et representation du groupe symetrique, Strasbourg, 1976.

16. S. Zaks, "Lexicographic generation of ordered trees," Theoret. Comput. Sci., 10 (1980), 63-82.

17. S. Zaks and D. Richards, "Generating trees and other combinatorial objects lexicographically," SIAM J. Comput., 8 (1979), 73-81.

18. S. Zaks, "Generation and ranking of k-ary trees," Info. Proc. Lett., 14 (1982), 44-48.

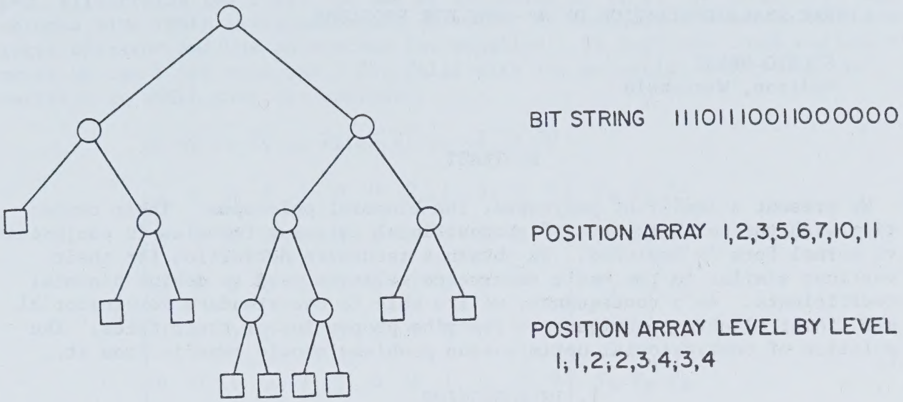

BIT STRING 1110111001100000
POSITION ARRAY 1,2,3,5,6,7,10,11
POSITION ARRAY LEVEL BY LEVEL
1; 1,2; 2,3,4; 3,4

Fig. 1

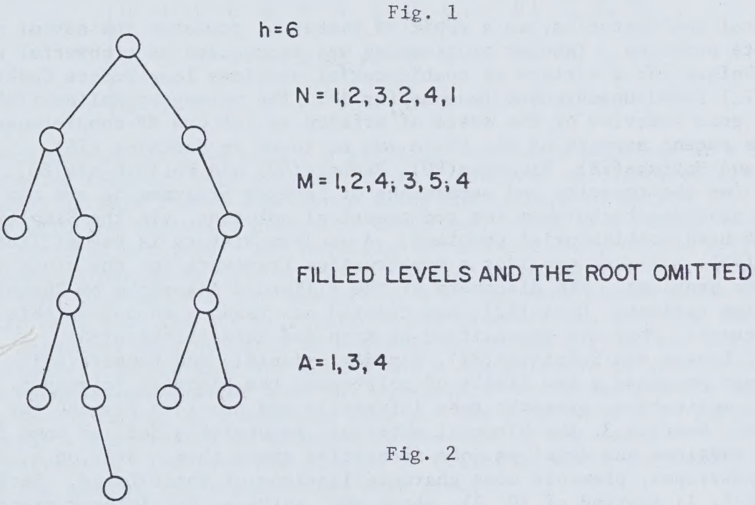
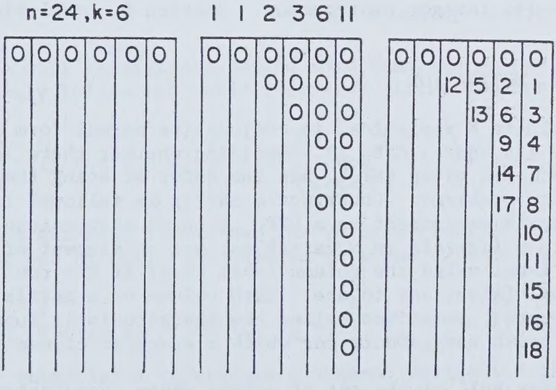

h=6
N = 1, 2, 3, 2, 4, 1
M = 1, 2, 4; 3, 5; 4
FILLED LEVELS AND THE ROOT OMITTED
A = 1, 3, 4

Fig. 2


n=24, k=6
1 1 2 3 6 11
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0
0 0 0
0 0
0 0
0 0
0
0
0
0
0
0 0 0 0 0 0
12 5 2 1
13 6 3
9 4
14 7
17 8
10
11
15
16
18

(a) (b) (c)

Fig. 3

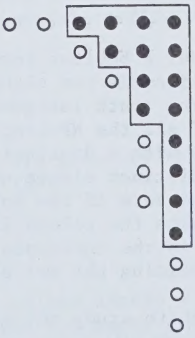


Fig. 4

A LINEAR CHARACTERIZATION OF NP-COMPLETE PROBLEMS

SILVIO URSIC
Madison, Wisconsin

ABSTRACT

We present a family of polytopes, the binomial polytopes. Their connection with the solution sets of propositional calculus formulas in conjunctive normal form is explored. We obtain a recursive definition for their vertices similar to the basic recurrence relation used to define binomial coefficients. As a consequence, we are able to use standard combinatorial and linear algebra techniques to describe properties of their faces. The solution of combinatorial optimization problems should benefit from it.

1. INTRODUCTION

Polyhedral combinatorics, as a topic of research, predates the advent of NP-complete problems. Integer programming was recognized as a powerful unifying technique for a variety of combinatorial problems long before Cook(71) and Karp(72) found unambiguous reasons for it. The survey by Balinski(65) is a very good overview of the state of affairs in the pre NP-completeness era. More recent surveys of the field can be found in Kastning (76), Balinski and Hoffman(78), Hausmann(80), Padberg(80) and Pulleyblank(83).

Reasons for the tenacity and permanence of integer programming are two fold. It provides techniques for the practical solution, via the Simplex Method, of hard combinatorial problems. A success history is exemplified by Grötschel(82). It also provides a constructive framework for the study of NP-complete problems. The discovery of the Ellipsoid Algorithm by Khachian (79) and its variants, Ursic(82), has created new reasons to pursue this line of attack. They are exemplified by Karp and Papadimitriou(80), Grötschel, Lovasz and Schrijver(81), Papadimitriou(81) and Lenstra(82).

This paper presents a new family of polytopes, the Binomial Polytopes. Section 2, motivation, presents them informally and provides reasons for their study. Section 3, the binomial matrices, recursively defines some families of matrices and develops some identities among them. Section 4, the binomial polytopes, presents some characterizations of their faces. Section 5, using (-1, 1) instead of (0, 1), shows why, perhaps, 0-1 integer programming should be changed to -1+1 integer programming. Section 6, conclusion, presents a final comment.

2. MOTIVATION

Consider a boolean formula in N variables, in conjunctive normal form and with p literals per clause (in short: $CNF_{N,p}$). Deciding whether there is a satisfying truth assignment to a given $CNF_{N,p}$ has the honor of being the first of all the NP-complete problems. Construct a matrix as follows: label each row with a distinct truth assignment to a $CNF_{N,p}$; label each column with a distinct clause with p literals in N variables; set an element of the matrix to zero if the row label makes the column label true; if the row label makes the column label false, set to one. Each column of a matrix so defined is the incidence vector, sometimes called the characteristic function, defining the set of truth assignments for which a specific clause is false.

We wish to study the convex hull of the set of points whose coordinates are given by the rows of the matrices defined above. Special consideration should be given to the case with p=3. The following example, with N=3 and p=2, illustrates the idea. We use as variables x, y, z. We label rows and columns with truth assignments and clauses. To indicate clauses we omit the logic operator and use an overbar for negation. To indicate truth assignments we use T for true and F for false with the subscript indicating the variable to which they are assigned.

$$\begin{array}{cccccccccccccc}
xy & x\bar{y} & \bar{x}y & \bar{x}\bar{y} & xz & x\bar{z} & \bar{x}z & \bar{x}\bar{z} & yz & y\bar{z} & \bar{y}z & \bar{y}\bar{z} & \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & F_x\ F_y\ F_z \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & F_x\ F_y\ T_z \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & F_x\ T_y\ F_z \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & F_x\ T_y\ T_z \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & T_x\ F_y\ F_z \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & T_x\ F_y\ T_z \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & T_x\ T_y\ F_z \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & T_x\ T_y\ T_z
\end{array} \qquad (2.1)$$

With matrix (2.1) we obtain a polytope with eight vertices in a space of dimension twelve. This polytope is in a subspace of dimension six. The removal of this linear dependence among the columns of matrix (2.1), together with the necessary change of variables, leads to a family of matrices, the binomial matrices. The knowledge of all the faces of the polytopes so defined allows us to test any $CNF_{3,2}$ for satisfiability. Add equations of the form "Clause" = 0, for example xy = 0, for each clause present in a specific $CNF_{3,2}$ to the set of equations and inequalities defining the polytope. The system so obtained will be satisfiable if and only if our $CNF_{3,2}$ is satisfiable. Notice that for each N and p the polytope in question is fixed. Its vertices are labeled with all possible 2^N truth assignments and its faces do not depend on a specific $CNF_{N,p}$. The purpose of this paper is to provide a description as accurate as possible of the faces of this family of polytopes. The task was started in Ursic(75) where the family was defined and some of their properties were given.

3. THE BINOMIAL MATRICES

We will recursively define some families of matrices. The definitions closely follow the basic recursive definition for binomial coefficients, namely:

$$\binom{0}{0} = 1;\ \text{for } i \neq 0, \binom{0}{i} = 0;\ \binom{N}{i} = \binom{N-1}{i-1} + \binom{N-1}{i}. \qquad (3.1)$$

The building blocks for the recursive definitions are given by the following one by one, with labeled rows and columns, matrices:

$$a_x = \overset{\phi}{[1]}\ T_x;\ b_x = \overset{x}{[0]}\ T_x;\ c_x = \overset{\phi}{[1]}\ F_x;\ d_x = \overset{x}{[1]}\ F_x. \qquad (3.2)$$

The subscript x represents a logical variable. The column labels ϕ and x represent clauses of a $CNF_{1,1}$. The label ϕ represents the empty clause. The row labels T_x and F_x are the two truth assignments to a $CNF_{1,1}$. We will use + and . to indicate standard matrix sums and products. We will use

$\oplus$ ($\oplus^T$) to indicate the row-wise (column-wise) join, or concatenation, of two matrices with the same number of identically labeled rows (columns). All row and column labels are retained in the result. We will use $\otimes$ to indicate standard tensor product of two matrices. Row and column labels of the result are sets of row and column labels of the factors. See, for example, Pease(65). In small examples we will use as subscripts x, y, z, w. In recursive definitions we will use $x_1, x_2, \ldots, x_N$, and write, for example, a_i instead of a_{x_i}.

Let us define some families of matrices:

$$E_i = a_i \oplus^T c_i = \begin{array}{cc} \phi & \\ \left[\begin{matrix} 1 \\ 1 \end{matrix}\right] & \begin{matrix} T_i \\ F_i \end{matrix} \end{array} \; ; \; V_i = b_i \oplus^T d_i = \begin{array}{cc} x_i & \\ \left[\begin{matrix} 0 \\ 1 \end{matrix}\right] & \begin{matrix} T_i \\ F_i \end{matrix} \end{array} \; ; \tag{3.3}$$

$$G_i = E_i \oplus V_i = \begin{array}{cc} \begin{matrix} \phi & x_i \end{matrix} & \\ \left[\begin{matrix} 1 & 0 \\ 1 & 1 \end{matrix}\right] & \begin{matrix} T_i \\ F_i \end{matrix} \end{array} = (a_i \oplus b_i) \oplus^T (c_i \oplus d_i); \tag{3.4}$$

$$r_{0,0,0} = 1; \text{ for } i, j \neq 0,\ r_{0,i,j} = \phi;\ r_{N,i,j} =$$
$$(r_{N-1,i,j} \otimes a_N \oplus r_{N-1,i,j-1} \otimes b_N) \oplus^T (r_{N-1,i-1,j} \otimes c_N \oplus r_{N-1,i-1,j-1} \otimes d_N); \tag{3.5}$$

$$R_{N,j} = \bigoplus^T_{0 \le i \le N} r_{N,i,j} \; ; \tag{3.6}$$

$$B_{N,p} = \bigoplus_{0 \le j \le p} R_{N,j} = \bigoplus_{0 \le j \le p} \Big(\bigoplus^T_{0 \le i \le N} r_{N,i,j} \Big). \tag{3.7}$$

The matrices $B_{N,p}$ are the Binomial Matrices. From (3.3), (3.5), (3.6) and (3.7) we obtain:

$$R_{0,0} = 1; \text{ for } i \neq 0,\ R_{0,i} = \phi;$$
$$R_{N,i} = R_{N-1,i-1} \otimes V_N \oplus R_{N-1,i} \otimes E_N; \tag{3.8}$$

$$\text{for } i \ge 0,\ B_{0,i} = 1; \text{ for } i < 0,\ B_{0,i} = \phi;$$
$$B_{N,i} = B_{N-1,i-1} \otimes V_N \oplus B_{N-1,i} \otimes E_N. \tag{3.9}$$

The symbol ϕ stands for the empty matrix.

Using definitions (3.4) and (3.5) the next theorem presents our basic matrix identity. It can be considered a generalization of the binomial identity

$$\sum_i \binom{N}{i} = 2^N . \tag{3.10}$$

Its proof closely follows a proof by induction of (3.10), applied twice to the indices i and j of definition (3.5). The matrices $r_{N,i,j}$ have $\binom{N}{i}$ rows and $\binom{N}{j}$ columns and behave like binomial coefficients either by rows or by columns. The matrices G_i perform the role of the constant two in (3.10). The matrix identity of theorem B is also an identity for the respective row and column labels.

Theorem B (The binomial identity). The following identity holds:

$$\bigotimes_{1 \le i \le N} G_i = \bigoplus^{T}_{0 \le i \le N} \Big(\bigoplus_{0 \le j \le N} r_{N,i,j} \Big) . \tag{3.11}$$

Proof. The proof is by induction on N. For N = 1, identity (3.11) becomes $G_1 = (a_1 \oplus b_1) \oplus^T (c_1 \oplus d_1)$, which is true by definition. The induction step is taken care as follows:

$$\begin{aligned}
\bigotimes_{1 \le i \le N} G_i &= \Big(\bigotimes_{1 \le i \le N-1} G_i \Big) \otimes G_N \\
&= \Big(\bigoplus^{T}_{0 \le i \le N-1} \Big(\bigoplus_{0 \le j \le N-1} r_{N-1,i,j} \Big) \Big) \otimes \big((a_N \oplus b_N) \oplus^T (c_N \oplus d_N) \big) \\
&= \bigoplus^{T}_{0 \le i \le N-1} \Big(\bigoplus_{0 \le j \le N-1} \big((r_{N-1,i,j} \otimes a_N \oplus r_{N-1,i,j} \otimes b_N) \oplus^T \\
&\qquad\qquad (r_{N-1,i,j} \otimes c_N \oplus r_{N-1,i,j} \otimes d_N) \big) \\
&= \bigoplus^{T}_{0 \le i \le N} \Big(\bigoplus_{0 \le j \le N} \big((r_{N-1,i,j} \otimes a_N \oplus r_{N-1,i,j-1} \otimes b_N) \oplus^T \\
&\qquad\qquad (r_{N-1,i-1,j} \otimes c_N \oplus r_{N-1,i-1,j-1} \otimes d_N) \big) \\
&= \bigoplus^{T}_{0 \le i \le N} \Big(\bigoplus_{0 \le j \le N} r_{N,i,j} \Big) . \quad \square
\end{aligned}$$

From definition (3.7) and identity (3.11) we derive the following identity, which provides an alternate description for $B_{N,N}$.

$$B_{N,N} = \bigotimes_{1 \le i \le N} G_i . \tag{3.12}$$

These two descriptions for $B_{N,N}$ also give us two distinct row and column orderings for it. The matrix $B_{N,N}$ is triangular in both of them.

$B_{3,3}$ in tensor product order:

$$\begin{array}{cccccccccc}
\phi & x & y & xy & z & xz & yz & xyz & \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T_xT_yT_z \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & F_xT_yT_z \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & T_xF_yT_z \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & F_xF_yT_z \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & T_xT_yF_z \\
1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & F_xT_yF_z \\
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & T_xF_yF_z \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & F_xF_yF_z
\end{array}$$

$B_{3,3}$ in binomial order:

$$\begin{array}{cccccccccc}
\phi & x & y & z & xy & xz & yz & xyz & \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T_xT_yT_z \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & F_xT_yT_z \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & T_xF_yT_z \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & T_xT_yF_z \\
1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & F_xF_yT_z \\
1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & F_xT_yF_z \\
1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & T_xF_yF_z \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & F_xF_yF_z
\end{array}$$

Triangularity of the tensor product order follows immediately from the fact that the matrices G_i are triangular. Triangularity of the binomial order can be shown by induction on N. We use (3.5) and show that $r_{N,i,i}$ = the identity matrix, for i < j, $r_{N,i,j}$ = a matrix of zeros.

As $B_{N,N}$ is full rank, so will be the matrices $B_{N,p}$. In fact, the columns of $B_{N,p}$ are a subset of the columns of $B_{N,N}$. It can be shown that the columns of $B_{N,p}$ are a basis for the column space of the matrices whose construction was outlined in section 2. The search for a suitable column space basis for them was the original motivating factor which led to the binomial matrices. In identities, matrices $B_{N,p}$ behave similarly to the sum:

$$\sum_{0 \le i \le P} \binom{N}{i} = \binom{N}{0} + \binom{N}{1} + \binom{N}{2} + \ldots + \binom{N}{p}. \tag{3.13}$$

Knuth(68) on page 64 mentions that there seems to be no simple formula for (3.13), meaning it does not seem to be possible to obtain identities involving (3.13) without the sum sign. It is nevertheless possible to derive meaningful identities involving (3.13). Riordan(68), pp 128-130, reports some results in this direction. The matrix identity involving $B_{N,p}$ needed for our purposes is the analogue of

$$\sum_{0 \le i \le p} \binom{N+M}{i} = \sum_{0 \le i \le N} \binom{N}{i} \sum_{0 \le j \le p-i} \binom{M}{j} = \sum_{0 \le i+j \le p} \binom{N}{i}\binom{M}{j} \tag{3.14}$$

namely:

$$B_{N+M,p} = \bigoplus_{0 \le i \le N} (R_{N,i} \otimes B_{M,p-i}) = \bigoplus_{0 \le i+j \le p} (R_{N,i} \otimes R_{M,j}). \tag{3.15}$$

Identity (3.14) can be proved by induction on N. The necessary index manipulations closely follow the corresponding ones in a proof by induction of (3.15). This concludes our study of "Binomial Matrixology". It would be interesting, and useful, to precisely characterize which binomial coefficient identities can be converted into binomial matrices identities.

4. THE BINOMIAL POLYTOPES

The binomial polytopes are the convex hull of the 2^N points whose coordinates are given by the rows of the matrices $B_{N,p}$. Theorem B informs us that the matrix $B_{N,N}$ has full rank. Therefore polytope $B_{N,N}$ is a simplex of dimension 2^N-1 in a space of dimension 2^N. To obtain a simplex whose dimension matches the space dimension we have to eliminate one of the columns of $B_{N,N}$. The column of $B_{N,N}$ with the empty label (matrix $R_{N,0}$) performs, very nicely, the role of the homogenization column. We will keep this column in the binomial matrices with the understanding that to obtain descriptions of our convex hulls we must add the equality $\phi = 1$ to the system of inequalities describing faces.

The inverse of matrix $B_{N,N}$ is easy to compute. From (3.12) we obtain:

$$B^{-1}_{N,N} = \bigotimes_{1 \le i \le N} G_i^{-1} = \bigotimes_{1 \le i \le N} \begin{array}{cc} T_i & F_i & \\ \left[\begin{array}{c} 1 \\ -1 \end{array}\right. & \left.\begin{array}{c} 0 \\ 1 \end{array}\right] & \begin{array}{c} \phi \\ x_i \end{array} \end{array}. \tag{4.1}$$

Identity (4.1) tells us what the faces of $B_{N,N}$ are. Each column of $B^{-1}_{N,N}$ describes a face. The polytopes $B_{N,p}$ are projections of $B_{N,N}$, hence it is possible to obtain their faces by Fourier elimination on the (known) faces of $B_{N,N}$.

Let $f_{N,p}$ be a column vector whose element labels match the column labels of $B_{N,p}$. Also, by definition, $h_{N,p} = B_{N,p} \cdot f_{N,p}$. Let $F(B_{N,p}, h_{N,p})$ be the submatrix of $B_{N,p}$ constructed with the rows of $B_{N,p}$ for which the corresponding components of $h_{N,p}$ are equal to zero. The vector $f_{N,p}$ describes a face of $B_{N,p}$ if:

(I) $h_{N,p} \geq 0$ and
(II) $F(B_{N,p}, h_{N,p})$ after removal of the homogenization column, has full rank. (4.2)

As $B_{N,p}$ consists of a subset of the columns of $B_{N,N}$, it is profitable to define the column vector $f^*_{N,p}$ as follows:

$$f^*_{N,p} = B^{-1}_{N,N} \cdot h_{N,p} = B^{-1}_{N,N} \cdot B_{N,p} \cdot f_{N,p}.$$

The vector $f^*_{N,p}$ represents the padded version of $f_{N,p}$, the padding being zeros added for all the columns of $B_{N,N}$ not present in $B_{N,p}$. It is convenient to explicitly list the subscripts of the matrices used to generate $B_{N,p}$. We write: $B_{3,3}(x, y, z) = G_x \otimes G_y \otimes G_z$. We proceed analogously for faces. We write $f_{4,3}(x, y, z, w)$ for a face of $B_{4,3}(x, y, z, w)$. We are now ready to start to describe faces of the polytopes $B_{N,p}$.

Theorem R (Representative faces). Let $[-1]_N$ and $[1]_\phi$ be one by one matrices with row labels x_N and ϕ respectively, and an empty column label. Then, for N = p or p even, the recurrence

$$\text{for } i \geq 0,\ f_{0,i} = [1]_\phi;\ \text{for } i < 0,\ f_{0,i} = \phi;$$

$$f_{N,p} = f_{N-1,p-1} \otimes [-1]_N \ \oplus^T \ f_{N-1,p} \otimes [1]_\phi \tag{4.3}$$

defines simplicial faces of $B_{N,p}$.

Proof. We will first verify condition (I) of (4.2). We have:

$$h_{N,p} = (B_{N-1,p-1} \otimes V_N \oplus B_{N-1,p} \otimes E_N) \cdot (f_{N-1,p-1} \otimes [-1]_N \oplus^T f_{N-1,p} \otimes [1]_\phi)$$

$$= (B_{N-1,p} \cdot f_{N-1,p}) \otimes E_N - (B_{N-1,p-1} \cdot f_{N-1,p-1}) \otimes V_N$$

$$= h_{N-1,p} \otimes E_N - h_{N-1,p-1} \otimes V_N \ . \tag{4.4}$$

Let us define $(h_{N,p})_q$ as a component of $h_{N,p}$ whose label contains exactly q occurrences of F. We wish to prove, by induction on N, that

$$(h_{N,p})_q = (-1)^p \cdot \binom{q-1}{p}. \tag{4.5}$$

As $0 \leq q \leq N$, if $N = 0$ then $q = 0$. Hence, for $N = 0$ (and $p \geq 0$):

$$(h_{0,p})_0 = (-1)^p \cdot \binom{-1}{p} = 1 \ . \tag{4.6}$$

As $B_{0,p} \cdot f_{0,p}$ assumes the same value as (4.6), we have that (4.5) is true for $N = 0$. Now for the induction step, which has two cases.

Case 1. Components of $h_{N,p}$ whose label contains T_N. In this case, from (4.4) we have that $(h_{N,p})_q = (h_{N-1,p})_q$ and (4.5) follows.

Case 2. Components whose label contains F_N. In this case, also from (4.4), $(h_{N,p})_q = (h_{N-1,p})_{q-1} - (h_{N-1,p-1})_{q-1}$ or, using the induction hypothesis $(-1)^p \cdot \binom{q-1}{p} = (-1)^p \cdot \binom{q-2}{p} - (-1)^{p-1} \cdot \binom{q-2}{p-1}$. (4.7)

Formula (4.7) is **the basic** binomial identity, namely (3.1). Hence (4.5) holds.

If p even and $q \geq 0$, we have $(-1)^p \cdot \binom{q-1}{p} \geq 0$.

If $p = N$ then $q-1 < p$ and (4.5) is equal to zero, unless $q = 0$, in which case (4.6) holds and (4.5) is equal to one.

We have verified condition (I) of (4.2). For condition (II) of (4.2), note that $(h_{N,p})_q = 0$ for $1 \leq q \leq p$. Hence:

$$F(B_{N,p}, h_{N,p}) = \bigoplus_{1 \leq i \leq p}^{T} \Big(\bigoplus_{0 \leq j \leq p} r_{N,i,j} \Big). \tag{4.8}$$

By theorem B and the triangularity of the binomial order it follows that matrix(4.8) satisfies condition(II) of (4.2). It follows that the faces of $B_{N,p}$ defined by (4.3) are simplicial. □

The next result formalizes the fact that the association of zero with true and one with false is arbitrary. The underlying idea consists in interchanging zeros and ones with full rank linear operators. To do so, we need some additional two by two matrices, namely:

$$I_x = \begin{array}{ccl} \phi & x & \\ 1 & 0 & \phi \\ 0 & 1 & x \end{array} ; \quad J_x = \begin{array}{ccl} T_x & F_x & \\ 1 & 0 & T_x \\ 0 & 1 & F_x \end{array} ; \quad Q_x = \begin{array}{ccl} \phi & x & \\ 1 & 1 & \phi \\ 0 & -1 & x \end{array} ; \quad P_x = \begin{array}{ccl} T_x & F_x & \\ 0 & 1 & T_x \\ 1 & 0 & F_x . \end{array} \tag{4.9}$$

The following identites hold: $Q_xQ_x = I_x$ and so $(Q_x)^{-1} = Q_x$; $P_xP_x = J_x$ and so $(P_x)^{-1} = P_x$; $P_xG_xQ_x = G_x$ and so $Q_x = G_x^{-1}P_xG_x$. With matrices (4.9) we define two linear operators:

$$P_{N,i} = J_1 \otimes J_2 \otimes \dots \otimes P_i \otimes \dots \otimes J_N; \quad Q_{N,i} = I_1 \otimes I_2 \otimes \dots \otimes Q_i \otimes \dots \otimes I_N.$$

We immediately obtain the identity $P_{N,i} \cdot B_{N,N} \cdot Q_{N,i} = B_{N,N}$.

Theorem N(Negation of variables). If $f^*_{N,p}(x_1, \dots, x_N)$ is a face of $B_{N,p}$ then so is $Q_{N,i} \cdot f^*_{N,p}(x_1, \dots, x_N) = f^*_{N,p}(x_1, \dots, \bar{x}_i, \dots, x_N)$.

Proof. We have $h_{N,p} = B_{N,N} \cdot f^*_{N,p}$, hence

$$P_{N,i} \cdot h_{N,p} = P_{N,i} \cdot B_{N,N} \cdot (Q_{N,i} \cdot Q_{N,i}) \cdot f^*_{N,p}$$

$$= (P_{N,i} \cdot B_{N,N} \cdot Q_{N,i}) \cdot Q_{N,i} \cdot f^*_{N,p} = B_{N,N} \cdot (Q_{N,i} \cdot f^*_{N,p}).$$

As $h_{N,p} \geq 0$ and the fact that $P_{N,i}$ is a permutation matrix we conclude that $P_{N,i} \cdot h_{N,p} \geq 0$. Hence $Q_{N,i} \cdot f^*_{N,p}$ satisfies the inequality condition, namely (I) of (4.2), to be a face. Condition (II) of (4.2) follows from: $P_{N,i}$ and $Q_{N,i}$ have full rank; analogously to the result obtained in theorem B, matrix $Q_{N,i}$ is upper triangular when permuted in binomial order. Hence the submatrix of $Q_{N,i}$ corresponding to the variables present in $B_{N,p}$ (its column labels) is also upper triangular and hence full rank. □

The next result expands considerably the characterization of faces of binomial polytopes. It has been known for quite some time that polytopes associated with hard combinatorial problems have relatively few vertices in relation to their very large number of faces. In other words, they are strongly degenerated. At the end of Gomory(64) we find an interesting discussion of some of the earlier work on the subject. A small example associated with the traveling salesman problem is characterized by H. W. Kuhn as a "miserable polyhedron". How can a polyhedron with so few vertices have so many faces? In the next page J. Edmonds remarks that as the vertices have a simple description, so might its faces. The following theorem gives some hope that a complete description of the faces of a polytope associated with the set of solutions to an NP-complete problem might indeed be possible.

Theorem C (Composition of faces). Let $g^*_{M,q}$ be a face of $B_{M,q}$ as given either by this theorem or by theorem R. Let $f^*_{N,p}$ be a face of $B_{N,p}$ as given by theorem R. Then $g^*_{M+N,p+q} = g^*_{M,q} \otimes f^*_{N,p}$ is a face of $B_{N+M,p+q}$.

Proof. First we check (I) of (4.2). We have

$$(B_{N,N} \cdot f^*_{N,p}) \otimes (B_{M,M} \cdot g^*_{M,q}) = (B_{N,N} \otimes B_{M,M}) \cdot (f^*_{N,p} \otimes g^*_{M,q})$$

$$= B_{N+M,N+M} \cdot g^*_{N+M,p+q} .$$

As $B_{N,N} \cdot f^*_{N,p} \geq 0$ and $B_{M,M} \cdot g^*_{M,q} \geq 0$ it follows that $B_{N+M,N+M} \cdot g^*_{N+M,p+q} \geq 0$. Hence $g^*_{M+N,p+q}$ satisfies (I) of (4.2).

The proof for (II) of (4.2) is complicated by the fact that $B_{N,p} \otimes B_{M,q} \neq B_{N+M,p+q}$. We must check the rank of $F(B_{N+M,p+q}, (B_{N,N} \cdot f^*_{N,p}) \otimes (B_{M,M} \cdot g^*_{M,q}))$. To this end, let us inspect the structure of $B_{N+M,p+q}$ as given by identity (3.15). Identity (3.15) tells us that from $B_{N,N}$ we obtain matrix $B_{N+M,p+q}$ by "substituting" each of its nonzero entries with one of the matrices $B_{M,p+q-i}$. Hence $B_{N+M,p+q}$ in the row and column order induced by the binomial order on $B_{N,N}$ is a block triangular matrix. The diagonal blocks are the matrices, for $0 \leq i \leq N$, $B_{M,p+q-i}$.

The submatrix of $B_{N+M,p+q}$ corresponding to the vertices touched by $g_{N+M,p+q}$ is also block triangular. Its diagonal blocks are:

$$\begin{aligned} &\text{for } 1 \leq i \leq p,\ B_{M,p+q-i}; \\ &\text{for } i = 0 \text{ and } p < i \leq N,\ F(B_{M,p+q-i}, B_{M,M} \cdot g^*_{M,q}). \end{aligned} \tag{4.10}$$

This follows from the fact that $f^*_{N,p}$ is a face of $B_{N,p}$ as given by theorem R. We will check the rank of these diagonal blocks. For $1 \leq i \leq p$, by theorem B, the matrices $B_{M,p+q-i}$ have full rank. For $i > p$, the matrices in (4.10) also have full rank. To verify this, first rewrite them, shifting the indexing, as: for $j \geq 1$, $F(B_{M,q-j}, B_{M,M} \cdot g^*_{M,q})$. This matrix is a subset of the columns of $F(B_{M,q}, B_{M,q} \cdot g_{M,q})$ and, as $g_{M,q}$ is a face of $B_{M,q}$, is full rank. One more of the diagonal blocks in (4.10) remains to be checked, namely the one with $i = 0$.

Consider the submatrix of $B_{N+M,p+q}$ given by: (Z is a matrix of zeros)

$$(F(B_{M,p+q}, B_{M,M} \cdot g^*_{M,q}) \oplus Z) \oplus^T (B_{M,p+q} \oplus B_{M,q}). \tag{4.11}$$

Transform it, with a linear combination of rows, into:

$$(Z \oplus F(B_{M,q}, B_{M,M} \cdot g^*_{M,q})) \oplus^T (B_{M,p+q} \oplus B_{M,q}) . \tag{4.12}$$

But, as $g_{M,q}$ is a face of $B_{M,q}$, the matrix $F(B_{M,q}, B_{M,M} \cdot g^*_{M,q})$, with its homogenization column removed, is full rank. Also, $B_{M,p+q}$ is full rank. Hence (4.11), with its homogenization column removed, is full rank. We conclude that condition (II) of (4.2) is verified. □

The next theorem formalizes the link between a $B_{N,p}$ and a $CNF_{N,p}$, outlined in section 2. It also shows how to use binomial polytopes to solve discrete optimization problems.

Theorem L (The linear characterization). Each clause of a $CNF_{N,p}$ has a corresponding face of $B_{N,p}$ whose vertices are labeled with all the truth assignments that make the clause true.

Proof. Let $x_1, x_2, \ldots, x_p$ be the literals in our clause. The corresponding face of $B_{N,p}$ is $f_{1,1}(x_1) \otimes f_{1,1}(x_2) \otimes \ldots \otimes F_{1,1}(x_p)$. The fact that it is indeed a face of $B_{N,p}$ follows from theorems R, N, and C. It is straightforward to verify that this face touches the required vertices of the polytope $B_{N,p}$. $\square$

As a final example we describe in terms of theorems R, N, and C all the faces of $B_{4,3}(x, y, z, w)$. Its Gale diagram is (1, 1, 1, 1, 1, 1, 1, 1, -1, -1, -1, -1, -1, -1, -1,-1). Consult Stoer & Witzgall(70) for an inspired introduction to Gale diagrams. Hence $B_{4,3}$ has 64 combinatorially isomorphic faces. They can be divided into two classes:

Type I; $f_{1,1} \otimes f_{1,1} \otimes f_{1,1} \otimes f_{1,0}$; Type II, $f_{3,2} \otimes f_{1,1}$.

Each type contains 32 faces. We list a representative set of 8 for each.

Type I:

$$1 - x - y - z + xy + xz + yz - xyz \geq 0 \;;$$
$$x - xy - xz + xyz \geq 0 \;;$$
$$y - xy - yz + xyz \geq 0 \;;$$
$$z - xz - yz + xyz \geq 0 \;;$$
$$xy - xyz \geq 0 \;;$$
$$xz - xyz \geq 0 \;;$$
$$yz - xyz \geq 0 \;;$$
$$xyz \geq 0.$$

Type II:

$$1 - x - y - z - w + xy + xz + xw + yz + yw + zw - xyw - xzw - yzw \geq 0;$$
$$x - xy - xz - xw + yz + xyw + xzw - yzw \geq 0;$$
$$y - xy + xz - yz - yw + xyw - xzw + yzw \geq 0;$$
$$z + xy - xz - yz - zw - xyw + xzw + yzw \geq 0;$$
$$w - xw - yw - zw + xyw + xzw + yzw \geq 0;$$
$$xw - xyw - xzw + yzw \geq 0;$$
$$yw - xyw + xzw - yzw \geq 0;$$
$$zw + xzw - xzw - yzw \geq 0.$$

5. USING (-1, 1) INSTEAD OF (0, 1)

The matrix $B_{N,N}$ is triangular and the associated polytope is a simplex. It is not, however, a regular simplex. Not much changes in our exposition if we define the entry in b_x of (3.2) as -1 instead of zero. With this choice, the matrix $B_{N,N}$ becomes orthogonal and the associated polytope is a regular simplex. The matrices $B_{N,p}$ exhibit a nice symmetry. In this (-1, 1) form, matrix $B_{N,N}$ has been known a long time, since the work of Sylvester(1867). The fact that these matrices have such a simple inverse, namely (4.1), is the basis of all the fast transform algorithms, following Good (58). Andrews & Kane(70) exposition makes clear the generality of the technique.

6. CONCLUSION

We have presented a family of polytopes, the Binomial Polytopes. Their connection with NP-complete problems has been explored. The Binomial Polytopes can be generated by a recurrence relation identical to the one with which we define binomial coefficients. As a consequence the vast body of knowledge accumulated in classical combinatorial analysis and linear algebra becomes available to the study of NP-completeness. Discrete optimization problems should benefit from it. The matrices $B_{N,p}$ can be described with Sylvester "decorative tile work". They are indeed interesting.

REFERENCES

1. H. C. Andrews, J. Kane, "Kronecker matrices, computer implementation, and generalized spectra", Journal of the ACM 17 (1970) 260-268.
2. M. L. Balinski, "Integer programming: methods, uses, computation", Management Science 12 (1965) 253-313.
3. M. L. Balinski, A. J. Hoffman, eds., "Polyhedral combinatorics", in: Mathematical programming study 8 (North-Holland, 1978) pp. 1-234.
4. S. A. Cook, "The complexity of theorem proving procedures", Proc. 3rd ACM STOC, (1971) 151-158.
5. R.E. Gomory, "The traveling salesman problem", in: Proc. IBM Scientific Computing Symposium on Combinatorial Problems, (1964) pp. 93-121.
6. I. J. Good, "The interaction algorithm and practical Fourier analysis", J. Royal Stat. Soc. Ser. B (1958) 361-372, ibid. (1960) 372-375.
7. M. Grötschel, L. Lovasz, A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization", Combinatorica 1 (1981) 169-197.
8. M. Grötschel, "Approaches to hard combinatorial optimization problems", in: B. Korte, ed., Modern Appl. Math., Opt. and O.R. (1982) 437-515.
9. D. Hausmann, ed., "Integer programming and related areas, a classified bibliography 1976-1978", (Springer Verlag, 1980).
10. R. M. Karp, "Reducibility among combinatorial problems", in: R. E. Miller J. W. Thacher, eds., Complexity of Comp. Co mput. (1972) 85-103.
11. R. M. Karp, C. H. Papadimitriou, "Onlinear characterizations of combinatorial optimization problems", IEEE 1980 FOCS pp. 1-9.
12. C. Kastning , ed., "Integer programming and related areas, a bibliography Lecture Notes in Economics and Mathematical Systems, (Springer Verl. 1976)
13. L. G. Khachian, " Apolynomial algorithm in linear programming", Soviet Mathematics Doklady 20 (1979) 191-194.
14. D. E. Knuth, "The art of computer programming",V.1 Addison-Wesley 1968.
15. H. W. Lenstra Jr., "Integer programming with a fixed number of variables" Report 81-03, University of Amsterdam, 1981.
16. M. W. Padberg ed., "Combinatorial optimization". in: Math. Progr. Study 12 (North-Holland 1980) pp. 1-221.
17. C. H. Papadimitriou, "On the complexity of integer programming", Journal of the ACM 28 (1981) 765-768.
18. W. R. Publeyblank, "Polyhedral combinatorics",in: Mathematical Programm ming, the State of the art, (Springer-Verlag 1983) pp.312-345.
19. J. Riordan, "Combinatorial Identities", (John Wiley & Sons 1968).
20. M. C. Pease III, "Methods of Matrix Algebra", (Academic Press 1965).
21. J. Stoer, C. Witzgall, "Convexity and Optimization in Finite Dimensions I", (Springer-Verlag 1970).
22. J. J. Sylvester, "Thoughts on inverse orthogonal matrices,simultaneous sign-successions,and tesselated pavements in two or more colours, with applications to Newton's rule, ornamental tile-work, and theory of numbers", Philosophical Magazine 34 (1867) 461-475.
23. S. Ursic, "A discrete optimization algorithm based on binomial polytopes" Ph. D. Thesis, University of Wisconsin, Madison, 1975.
24. S. Ursic, "Binomial Polytopes and NP-complete problems", in: J. F. Traub, ed., Algorithms and Complexity, New Directions and Recent Results, (Academic Press, 1976).
25. S. Ursic, "The ellipsoid algorithm for linear inequalites in exact arithmetic", IEEE 23 FOCS (1982) 321-326.
26. S. Ursic, "A linear characterization of NP-complete problems", submitted to: Mathematical Programming.

TIME-SEQUENTIAL SAMPLING IN N-DIMENSIONAL EUCLIDEAN SPACE*

J. P. Allebach
Purdue University, School of Electrical Engineering, West Lafayette, IN 47907

ABSTRACT

When sampling time-varying multidimensional signals, limited data collecting ability frequently constraints the sampling to be time-sequential. In this case, the order in which the spatial points are sampled has a significant effect on the resolution of spatiotemporal frequency components in the signal. Fourier analysis provides the basis for understanding this relationship. In particular, the congruential sampling pattern performs especially well. This pattern is found to have a number of interesting properties.

1. INTRODUCTION

Consider a process $g(\mathbf{x},t)$ which depends on N independent variables $\mathbf{x}=(x_1, \ldots, x_N)$, and time t. Suppose that we wish to take samples of this process as a function of these variables and time; but that we can only measure the process for one value of the vector **x** at a time. Thus the sampling is constrained to be time-sequential. Examples of such problems arise most commonly in imaging systems such as commercial broadcast television where the scene is scanned in two interlaced fields, each of which is sampled lexicographically, or infrared imaging systems that utilize a mosaic of sensors whose outputs are time-division multiplexed into one channel. In examples of this type, the vector **x** represents spatial position. However, in other cases, **x** may simply be a vector of parameters for some process that is evolving in time. While keeping this possibility in mind, we shall for convenience, always refer to **x** as the spatial coordinate of g.

Let's assume that we wish to sample each of the variables x_j at M_j points uniformly spaced in the interval $[0,A_j]$, $j = 1, \ldots, N$, and that we take one sample every T s. At time kT, we sample at $\mathbf{x} = (\alpha_{1k}X_1, \ldots, \alpha_{Nk}X_N)$, where $X_j = A_j/M_j$. The index vector sequence $\boldsymbol{\alpha}_k = (\alpha_{1k}, \ldots, \alpha_{Nk})$ defines the order in which the points are sampled. We shall assume that $\boldsymbol{\alpha}_k$ is a permutation of all N-tuples of integers $(a_1, \ldots, a_N)$, $0 \leq a_j \leq M_j-1$ for $k = 0, \ldots, [\mathbf{M}]_\bullet-1$, where $[\mathbf{M}]_\bullet = M_1 \cdots M_N$. Outside this range, we assume that the sequence repeats periodically; so $\boldsymbol{\alpha}_k = \boldsymbol{\alpha}_{k+b[\mathbf{M}]_\bullet}$ for integer b. We obtain an entire frame of samples in $B = [\mathbf{M}]_\bullet T$ s.

---

* This work was supported by the National Science Foundation under Grant ECS-8312231.

In some situations such as imaging with sensor arrays, any ordering α_k is permissible. In other situations, α_k may be partially constrained by hardware limitations. For example, electromechanical devices are often constrained to scanning lexicographically in one or more dimensions. Whether constrained or not, a lexicographic ordering is most commonly used. A number of researchers [1]-[5] have experimented with other orderings to achieve bandwidth compression in television systems. However, these results do not provide the basis for a complete understanding of time-sequential sampling. Fourier analysis has traditionally played an important role in problems of sampling and processing one- and multi-dimensional signals with various non-time-sequential sampling point geometries [6]-[9]. In this context, we analyzed time-sequential sampling on spatially rectangular and hexagonal lattices of 2 dimensions [10],[11]. The central result of this work was that in the frequency domain, time-sequential sampling causes replication of the spectrum of g on a lattice that is reciprocal to the sampling lattice, just as in the general multidimensional case [7]. Unlike the general case, however, the replications are individually weighted by coefficients that are the 3D discrete Fourier transform (DFT) of the sampling pattern. These coefficients control the extent of frequency domain aliasing, and are thus the key to sampling pattern performance.

The sampling patterns considered in the past were chosen on an *ad hoc* or experimental basis. Based on our analysis, we developed two methods for design of 2D sampling patterns [11],[12]. The first method uses a pairwise exchange algorithm to search for good patterns among those that have a certain structure imposed by mapping single indices to multiple indices. The second method searches for good patterns among those that can be generated by congruential multipliers. When g is not undersampled temporally by more than a factor of 2, the congruential sampling pattern is optimal for rectangular lattices in the sense that it maximizes the signal-to-aliasing noise power ratio at fixed spatial and temporal sampling rates. We obtain nearly identical performance from the patterns designed by both methods for either lattice which suggests that they are all nearly optimal. The congruential sampling patterns are of particular interest because of their simple structure.

The purpose of the present paper is threefold. First, we will extend our earlier analysis of time-sequential sampling on a 2D rectangular lattice to an arbitrary number of spatial dimensions N. Second, we will explore the properties of congruential sampling patterns in greater depth than we have so far. Finally, we will consider the effect of sampling ellipsoidally bandlimited signals with a congruential pattern.

2. POWER SPECTRUM OF THE SAMPLED SIGNAL AND NOISE POWER DUE TO ALIASING

We begin by establishing some notation that will simplify the mathematics. As we have so far, we will denote N-tuples of subscripted variables by a boldface version of the variable name. We shall apply the usual operations for scalars to these vectors with the understanding that the result will be a vector obtained by carrying out the operation element by element. For example, if $\mathbf{x} = (x_1, \ldots, x_N)$ and $\mathbf{y} = (y_1, \ldots, y_N)$, then $\mathbf{xy} = (x_1y_1, \ldots, x_Ny_N)$; $\mathbf{x}^{-1} = (x_1^{-1}, \ldots, x_N^{-1})$; and $\mathbf{x} \bmod \mathbf{y} = (x_1 \bmod y_1, \ldots, x_N \bmod y_N)$. Similarly, the expression $\mathbf{x} \leq \mathbf{y}$ expands to $x_1 \leq y_1, \ldots, x_N \leq y_N$; and

$$\sum_{\mathbf{a}=\mathbf{0}}^{\mathbf{M}-\mathbf{1}} = \sum_{a_1=0}^{M_1-1} \cdots \sum_{a_N=0}^{M_N-1} .$$

In addition, we treat multiplication of a vector by a scalar in the standard way, and define two operators that produce a scalar result from vector arguments: $\mathbf{x}\cdot\mathbf{y} = x_1y_1 + \cdots + x_Ny_N$, the usual inner product, and $[\mathbf{x}]_\bullet = x_1 \cdots x_N$. Note that $\mathbf{x}\cdot(\mathbf{yz})$ and $(\mathbf{x}\cdot\mathbf{y})\mathbf{z}$ produce different results; so the parentheses must be included.

Now, let $g(\mathbf{x},t)$ be a spatially and temporally wide-sense stationary process with zero mean and power spectral density $S_{gg}(\mathbf{u},f)$, where $\mathbf{u}$ and f are the spatial and temporal frequency variables. Although g does not have finite spatial support, we shall limit our interest to the region $\mathbf{0} \leq \mathbf{x} \leq \mathbf{A}$. Time-sequential sampling of g results in

$$h(\mathbf{x},t) = [\mathbf{X}]_\bullet B \sum_k g(\boldsymbol{\alpha}_k\mathbf{X},kT)\delta(\mathbf{x}-\boldsymbol{\alpha}_k\mathbf{X},t-kT). \tag{1}$$

Here $[\mathbf{X}]_\bullet B$ is the hypervolume of each sample cell; and $\delta(\mathbf{x},t)$ is the $(N+1)$-dimensional Dirac delta function. After randomizing the origin of h to make it wide-sense stationary, we find by straightforward extension of the approach used in [10] for $N = 2$, that the power spectrum of h is given by

$$S_{hh}(\mathbf{u},f) = \sum_{\mathbf{m}}\sum_{p} |Q_{\mathbf{m}p}|^2 S_{gg}(\mathbf{u}-\mathbf{m}\mathbf{A}^{-1}, f-pB^{-1}), \tag{2}$$

where

$$Q_{\mathbf{m}p} = 1/[\mathbf{M}]_\bullet \sum_{k=0}^{[\mathbf{M}]_\bullet-1} \exp\{-i2\pi[\mathbf{m}\cdot(\boldsymbol{\alpha}_k/\mathbf{M}) + pk/[\mathbf{M}]_\bullet]\}. \tag{3}$$

For any $\boldsymbol{\alpha}_k$, the coefficients satisfy

$$Q_{\mathbf{0}0} = 1, \tag{4}$$

$$Q_{\mathbf{m}0} = \delta_{\mathbf{m} \bmod \mathbf{M}}, \tag{5}$$

$$Q_{\mathbf{0}p} = \delta_{p \bmod [\mathbf{M}]_\bullet}, \tag{6}$$

$$Q_{\mathbf{m}+\mathbf{b}\mathbf{M},\, p+c[\mathbf{M}]_\bullet} = Q_{\mathbf{m}p}, \text{ for integer } \mathbf{b} \text{ and } c. \tag{7}$$

The functions δ_p and $\delta_\mathbf{m}$ are the one- and multi-dimensional Kronecker delta functions.

Let the support of $S_{gg}(\mathbf{u},f)$ be limited to a region Ω, and let $(\mathbf{U},W)$ be the dimensions of the smallest rectangular hyperparallelepiped that contains Ω. From (5) and (7), it follows that the replications of S_{gg} with nonzero weight in the $p=0$ hyperplane of (2) that are closest to the origin are located at $(\mathbf{u},f) = (X_1^{-1},0,\ldots,0),(0,X_2^{-1},0,\ldots,0),\ldots,(0,\ldots,0,X_N^{-1})$. The replications in the $p=\pm1$ hyperplanes are located at least B^{-1} from the origin. Therefore, no replications will overlap the baseband term at $(\mathbf{0},0)$ if

$$\mathbf{U} \leq \mathbf{X}^{-1}/2, \tag{8}$$

$$W \leq B^{-1}/2. \tag{9}$$

These are the worst case Nyquist conditions for time-sequential sampling in N spatial dimensions. Because the weighting of the replications in the $p=0$ hyperplane does not depend on $\boldsymbol{\alpha}_k$, the spatial Nyquist rate is the same for all sampling patterns. However, the temporal Nyquist rate may be less than 2W if $Q_{\mathbf{m}p} = 0$ for values of $(\mathbf{m},p)$ other than those indicated in (6). This is the case, for example, with the congruential sampling pattern, as we shall see later.

If the Nyquist conditions are satisfied, g may be recovered from h with zero mean-squared error by low pass filtering h to the baseband region Ω. Otherwise, we recover g plus additive aliasing error

$$e(\mathbf{x},t) = \mathrm{LPF}_\Omega\{h(\mathbf{x},t)-g(\mathbf{x},t)\}, \tag{10}$$

which is uncorrelated with g and has power [10]

$$\sigma_e^2 = E\{|e(\mathbf{x},t)|^2\} = \sum_{(\mathbf{m},p)\neq(\mathbf{0},0)}\sum |Q_{\mathbf{m}p}|^2 V_{\mathbf{m}p}, \tag{11}$$

where

$$V_{\mathbf{m}p} = \iint_\Omega S_{gg}(\mathbf{u}-\mathbf{m}\mathbf{A}^{-1}, f-pB^{-1})d\mathbf{u}df. \tag{12}$$

Since the closest replications with nonzero weight in the $p=0$ hyperplane have weight unity for any sampling pattern, undersampling spatially will cause a rapid increase in the noise power σ_e^2. When undersampling temporally, the rate at which σ_e^2 increases will depend on the coefficients $Q_{\mathbf{m}p}$ and hence on the sampling pattern.

3. DERIVATION OF THE CONGRUENTIAL SAMPLING PATTERN

We solve for the congruential sampling pattern by minimizing σ_e^2 under the condition that the sampling be spatially at or above the Nyquist rate and temporally not more than a factor of 2 below the Nyquist rate; so (8) is satisfied, and $B^{-1} \geq W$. Following a procedure similar to that in [12] for $N=2$, (11) may be rewritten as

$$\sigma_e^2 = \sum_{\mathbf{m}=\mathbf{0}}^{\mathbf{M-1}} |Q_{\mathbf{m}1}|^2 Z_{\mathbf{m}1}, \tag{13}$$

where

$$Z_{\mathbf{m}p} = 2\sum_{\mathbf{b}=\mathbf{0}}^{\mathbf{1}} V_{\mathbf{m}-\mathbf{bM},p}\,. \tag{14}$$

The latter sum is over the 2^N N-tuples $\mathbf{b}$ containing only 0's and 1's.

So far, we have always specified sampling patterns in terms of the spatial coordinate $\boldsymbol{\alpha}_k$ where the k-th sample in the sequence is taken for $k = 0, \ldots, [\mathbf{M}]_\bullet - 1$. We may alternatively define the pattern in terms of the position $0 \leq \gamma_{\mathbf{a}} \leq [\mathbf{M}]_\bullet - 1$ of the spatial coordinate $\mathbf{a}$ in the time sequence for $\mathbf{a} = \mathbf{0}, \ldots, \mathbf{M-1}$. In this case, (3) becomes

$$Q_{\mathbf{m}p} = 1/[\mathbf{M}]_\bullet \sum_{\mathbf{a}=\mathbf{0}}^{\mathbf{M-1}} \exp\{-i2\pi[\mathbf{m}\cdot(\mathbf{a}/\mathbf{M}) + p\gamma_{\mathbf{a}}/[\mathbf{M}]_\bullet]\}. \tag{15}$$

For each value of p, this is an N-dimensional DFT with respect to $\mathbf{a}$ of the signal

$$q_{\mathbf{a}p} = \exp\{-i2\pi p\gamma_{\mathbf{a}}/[\mathbf{M}]_\bullet\}. \tag{16}$$

By Parseval's relation for the DFT,

$$\sum_{\mathbf{m}=\mathbf{0}}^{\mathbf{M-1}} |Q_{\mathbf{m}p}|^2 = 1, \text{ for each } p. \tag{17}$$

So an optimal sampling pattern will distribute 1 unit of energy over the coefficients $Q_{\mathbf{m}1}$ in a way that minimizes their weighted sum given by (13). Certainly, no pattern can do better than one which assigns all the energy to the smallest weight:

$$Q_{\mathbf{m}1} = \delta_{(\mathbf{m}-\hat{\mathbf{m}}) \bmod \mathbf{M}} \tag{18}$$

where $\hat{\mathbf{m}}$ satisfies

$$Z_{\hat{\mathbf{m}}1} \leq Z_{\mathbf{m}1}, \text{ for all } \mathbf{m}. \tag{19}$$

But does a sampling pattern exist with this $Q_{\mathbf{m}1}$? By straightforward extension of the approach in [12], we find that the answer to this question is "yes" if and only if

$$\gcd(\hat{\mathbf{m}}\overline{\mathbf{M}}, \mathbf{M}) = 1; \tag{20}$$

and we find that the sampling pattern is given by

$$\gamma_{\mathbf{a}} = -\hat{\mathbf{m}}\cdot(\mathbf{a}\overline{\mathbf{M}}) \bmod [\mathbf{M}]_\bullet, \tag{21}$$

or, equivalently

$$\boldsymbol{\alpha}_k = -(\hat{\mathbf{m}}\overline{\mathbf{M}})^{-1}k \bmod \mathbf{M}. \tag{22}$$

Here gcd denotes greatest common divisor, $\overline{\mathbf{M}} = [\mathbf{M}]_\bullet/\mathbf{M}$; and $(\hat{\mathbf{m}}\overline{\mathbf{M}})^{-1}$ is the inverse of $\hat{\mathbf{m}}\overline{\mathbf{M}}$ modulo $\mathbf{M}$; so there exists an integer N-tuple $\mathbf{b}$ such that $(\hat{\mathbf{m}}\overline{\mathbf{M}})^{-1}\hat{\mathbf{m}}\overline{\mathbf{M}} = \mathbf{1} + \mathbf{bM}$.

From (22), we see that the j-th coordinate of the k-th spatial sampling point in the sequence is obtained by multiplying k by a constant $-(\hat{m}_j\overline{M}_j)^{-1}$ and taking the product modulo the range of that coordinate. Thus, consecutive samples lie along a straight line that wraps around whenever a boundary of the spatial region is reached. The condition (20) ensures that the linear scan will not repeat until every point in the spatial region has been sampled once. The coefficients for congruential sampling patterns have a particularly simple structure as indicated by (18). Consequently, the noise power for these patterns under the conditions stated at the beginning of this section, is just

$$\sigma_e^2 = Z_{\hat{\mathbf{m}}1}, \tag{23}$$

Note that (20) implies $\gcd(M_j,M_k) = 1$, $j \neq k$; so the number of samples taken in each spatial dimension must form a set of relatively prime numbers. The value of $\hat{\mathbf{m}}$ that satisfies (19) will be a function of the sampling parameters $\mathbf{X}$ and B as well as the power spectrum S_{gg}. Depending on the complexity of the calculations for $Z_{\hat{\mathbf{m}}1}$ given by (12) and (14), it may be possible to solve for $\hat{\mathbf{m}}$ in closed form. Alternatively, we can simply search through the values that satisfy (20) to find the one that minimizes $Z_{\hat{\mathbf{m}}1}$.

4. THE DUAL CONGRUENTIAL SAMPLING PATTERN

The assumption in the previous section that $B^{-1} \geq W$ results in only the $p = \pm 1$ hyperplanes of (11) contributing to σ_e^2. In addition by symmetry, σ_e^2 can be expressed in terms of the $p = 1$ hyperplane only, as it is in (13). Thus, we specified the congruential sampling pattern in terms of $Q_{\mathbf{m}p}$ for just $p = 1$.

If the spatial Nyquist condition (8) is still satisfied, but $B^{-1} < W$, (13) becomes

$$\sigma_e^2 = \sum_{p=1}^{\infty} \sum_{\mathbf{m}=\mathbf{0}}^{\mathbf{M}-\mathbf{1}} |Q_{\mathbf{m}p}|^2 Z_{\mathbf{m}p}, \tag{24}$$

where $Z_{\mathbf{m}p}$ is given by (14) as before. Note that $Z_{\mathbf{m}p} = 0$ for $pB^{-1} \geq 2W$; so the inner sum does not actually range to ∞. Under these more general conditions, the coefficients $Q_{\mathbf{m}p}$ are of interest for $p > 1$. They can be derived straightforwardly by substituting (21) into (15); but it is more interesting to note from (16) that for any sampling pattern, the inverse DFT of $Q_{\mathbf{m}p}$ with respect to $\mathbf{m}$ satisfies

$$q_{\mathbf{a}p} = [q_{\mathbf{a}1}]^p. \tag{25}$$

By the convolution theorem,

$$Q_{\mathbf{m}p} = [Q_{\mathbf{m}1}]_{*p}, \tag{26}$$

where $[\,]_{*p}$ denotes p-fold self convolution.

For congruential sampling patterns, we may show directly by induction on (26) that

$$Q_{\mathbf{m}p} = \delta_{(\mathbf{m}-p\hat{\mathbf{m}}) \bmod \mathbf{M}} \ . \tag{27}$$

Thus we find that in each p-hyperplane, Q has value 1 at the coordinates

$$\tilde{\boldsymbol{\alpha}}_p = \hat{\mathbf{m}} p \bmod \mathbf{M}, \tag{28}$$

and value 0 elsewhere. Comparing $\tilde{\boldsymbol{\alpha}}_p$ with $\boldsymbol{\alpha}_k$ for the congruential sampling pattern given by (22), it is natural to ask whether $\tilde{\boldsymbol{\alpha}}_p$ is also a valid sampling pattern, *i.e.* is it a permutation of the N-tuples of integers $\mathbf{0} \leq \mathbf{a} \leq \mathbf{M}-\mathbf{1}$ for $p = 0, \ldots, [\mathbf{M}]_{\bullet}-1$?

To answer this question, we consider any p_1 and p_2 for which $\tilde{\boldsymbol{\alpha}}_{p_1} = \tilde{\boldsymbol{\alpha}}_{p_2}$. From (28), we have

$$\hat{\mathbf{m}}(p_1-p_2) = \mathbf{b}\mathbf{M}, \tag{29}$$

for some integer $\mathbf{b}$. Let $|$ denote "divides". For the j-th component of this vector equation, M_j $|$ the right side, and therefore $|$ the left side. From (20), $\gcd(\hat{m}_j \overline{M}_j, M_j) = 1$. Therefore, $\gcd(\hat{m}_j, M_j) = 1$; and $M_j \mid (p_1-p_2)$. Since this is true for all $1 \leq j \leq N$, $[\mathbf{M}]_{\bullet} \mid (p_1-p_2)$. This implies that $p_2 = p_1 \bmod [\mathbf{M}]_{\bullet}$, and hence that $\tilde{\boldsymbol{\alpha}}_p$ is, indeed, a valid sampling pattern.

We conclude that the coefficients $Q_{\mathbf{m}p}$ of any congruential sampling pattern $\boldsymbol{\alpha}_k$ form a second sampling pattern $\tilde{\boldsymbol{\alpha}}_p$ which is also congruential. The alternate form for this dual pattern is

$$\tilde{\gamma}_{\mathbf{m}} = (\hat{\mathbf{m}}\overline{\mathbf{M}})^{-1} \cdot (\mathbf{m}\overline{\mathbf{M}}) \bmod [\mathbf{M}]_{\bullet}. \tag{30}$$

To verify this, we multiply the above by $\hat{m}_j$ and then take everything modulo M_j. Repeating for $1 \leq j \leq N$, we obtain (28). Comparing (22) and (21) for $\boldsymbol{\alpha}_k$ and $\gamma_{\mathbf{a}}$ with (28) and (30) for $\tilde{\boldsymbol{\alpha}}_p$ and $\tilde{\gamma}_{\mathbf{m}}$, we see that with the exception of a sign change, the primal and dual sampling patterns are related by simply interchanging the roles of $\hat{\mathbf{m}}$ and $(\hat{\mathbf{m}}\overline{\mathbf{M}})^{-1}$.

In leaving this section, we note from (24) and (27) that with congruential sampling at or above the spatial Nyquist rate, the noise power is given by

$$\sigma_e^2 = \sum_{p=1}^{\infty} Z_{p\hat{\mathbf{m}} \bmod \mathbf{M}, p} \quad . \tag{31}$$

5. TEMPORAL NYQUIST RATE FOR ELLIPSOIDALLY BANDLIMITED SIGNALS SAMPLED WITH A CONGRUENTIAL PATTERN

In this section, we assume that the bandregion Ω is an (N+1)-dimensional hyperellipsoid; so

$$\Omega = \{(\mathbf{u}, f) \colon ||\mathbf{u}/\mathbf{U}||^2 + (f/W)^2 < 1\}\ , \tag{32}$$

where $||\mathbf{x}||^2 = \mathbf{x} \cdot \mathbf{x}$. With the congruential sampling pattern defined by (22), the aliasing noise power σ_e^2 is given by (31), (14), and (12). For $\sigma_e^2 = 0$, we must have $V_{p\hat{\mathbf{M}} \bmod \mathbf{M} - \mathbf{b}\mathbf{M}, p} = 0$; or equivalently,

$$\left\{ \| (p\hat{\mathbf{m}} \bmod \mathbf{M} - \mathbf{bM})/(\mathbf{A\,U}) \|^2 + [p/(BW)]^2 \right\}^{1/2} \geq 2 \,, \tag{33}$$

for all $\mathbf{b}$, $b_j = 0$ or 1, $1 \leq j \leq N$, and for all $p \geq 1$. With a fixed sampling pattern, the temporal Nyquist rate $B_{\min}^{-1}$ is the minimum value of B^{-1} for which this condition is satisfied. Rearranging (33) and assuming Nyquist spatial sampling so $\mathbf{A\,U} = \mathbf{M}/2$, we obtain

$$B_{\min}^{-1} = \max_{p \geq 1} \beta(p) \,, \tag{34}$$

where

$$\beta(p) = \max_{b_j=0,1} \frac{2W}{p} \left\{ \left[1 - \| (p\hat{\mathbf{m}} \bmod \mathbf{M})/\mathbf{M} - \mathbf{b} \|^2 \right]^+ \right\}^{1/2} \tag{35}$$

Here $[x]^+ = x$, if $x > 0$; and $[x]^+ = 0$, otherwise. Since $\mathbf{0} \leq (p\hat{\mathbf{m}} \bmod \mathbf{M})/\mathbf{M} \leq \mathbf{1}$, the maximum over $\mathbf{b}$ is attained when

$$\hat{b}_j = \begin{cases} 1, & (p\hat{m}_j \bmod M_j)/M_j > 1/2 \,, \\ 0, & \text{else.} \end{cases} \tag{36}$$

Then, $\left| (p\hat{\mathbf{m}} \bmod \mathbf{M})/\mathbf{M} - \hat{\mathbf{b}} \right| \leq \mathbf{1}/2$. We also note that $\beta(p) \leq 2W/p$. If we search for the maximum of $\beta(p)$ starting with $p = 1$, we can stop at p_0 if $\max\limits_{1 \leq p \leq p_0} \beta(p) \geq 2W/(p_0+1)$.

We have determined the temporal Nyquist rate for any fixed congruential sampling pattern. We are most interested, however, in finding the minimum Nyquist rate over all congruential patterns and identifying that pattern which achieves the minimum rate. From (36), we see that $\beta(1)$ is minimized with respect to $\hat{\mathbf{m}}$ when $\hat{\mathbf{m}}$ is as close to $\mathbf{M}/2$ as possible, subject to (20) and the fact that it must be integer-valued. It follows that $\beta(1) \simeq 2W\{[1-N/4]^+\}^{1/2}$. Since this quantity is $\geq W$ for $N = 1,2$, and 3, we conclude from the stopping rule stated above that $B_{\min}^{-1} = \beta(1)$ for these values of N. This approximation, which is in fact a lower bound for $B_{\min}^{-1}$, improves as $\mathbf{M}$ increases.

For $N \geq 4$, we have not determined a closed form expression for $B_{\min}^{-1}$; but for a specific $\mathbf{M}$, it is always possible to determine $B_{\min}^{-1}$ by searching over all permissible $\hat{\mathbf{m}}$ in the range $\mathbf{0} \leq \hat{\mathbf{m}} \leq \mathbf{M}-\mathbf{1}$. Figure 1 shows the result of repeating such a search as N is incremented from 1 to 8 and the spatial region is augmented each time with a new dimension containing M_N points. The lower bound for $N=1,2$, and 3 is also shown. The sampling rates are normalized to the worst case Nyquist rate of 2W. We see that the normalized sampling rate decreases steadily below 1.0 as N increases. Thus we can sample an ellipsoidally bandlimited signal without aliasing at a much lower rate using a congruential pattern instead of the more conventional lexicographic pattern for which the Nyquist rate is very close to the worst case value. For example, with 3 spatial dimensions the savings will approach 50%.

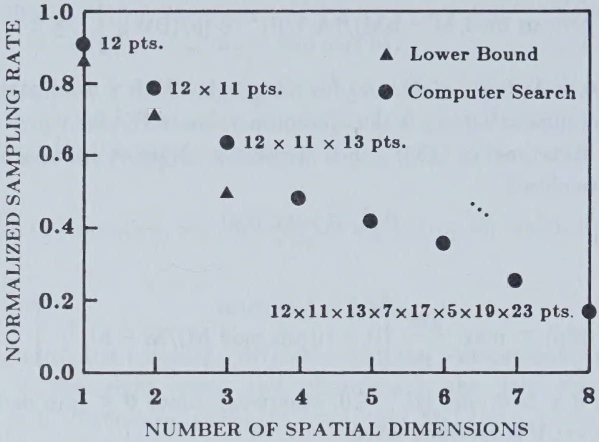


Fig. 1 Temporal Nyquist rate $B_{\min}^{-1}$ normalized by the worst case rate 2W as a function of the number of spatial dimensions N.

6. CONCLUSIONS

We have analyzed time-sequential sampling of time-varying signals with N spatial dimensions. The power spectrum of the sampled signal consists of replications of the power spectrum of the original signal on a lattice that is reciprocal to the sampling lattice. The replications are individually weighted by coefficients that are the (N+1)-dimensional DFT of the sampling pattern. With the congruential sampling pattern, these coefficients are a dual sampling pattern that is also congruential. Because of this structure, congruential sampling patterns appear to be especially attractive, both in terms of implementing signal processing systems and in terms of their analysis. For ellipsoidally bandlimited signals, a substantial reduction in the sampling rate is possible with a congruential sampling pattern.

REFERENCES

[1] S. Deutsch, "Pseudorandom dot scan television systems," *IEEE Trans. Broadcast,* vol. BC-11, pp. 11-21, July 1965.

[2] R. C. Brainard, F. W. Mounts, and B. Prasada, "Low resolution TV: subjective effects of frame repetition an picture replenishment," *Bell Syst. Tech. J.,* vol. 46, pp. 261-271, Jan. 1967.

[3] S. Deutsch, "Visual displays using pseudorandom dot scan," *IEEE Trans. Commun. Technol.,* vol. COM-21, pp. 65-75, Jan. 1973.

[4] R. F. Stone, "A practical narrow-band television system," *IEEE Trans. Broadcast.,* vol. BC-22, pp. 21-32, June 1976.

[5] K. Takikawa, "Simplified 6.3 Mbit/s codec for video conferencing," *IEEE Trans. Commun.,* vol. COM-29, pp. 1877-1882, Dec. 1981.

[6] A. J. Jerri, "The Shannon sampling theorem - its various extensions and applications: a tutorial review," *Proc. IEEE,* vol. 65, pp. 1565-1596, Nov. 1977.

[7] D. P. Petersen and D. Middleton, "Sampling and reconstruction of wave number-limited functions in N-dimensional Euclidean spaces," *Inf. Control,* vol. 5, pp. 279-323, Dec. 1962.

[8] R. M. Mersereau, "The processing of hexagonally sampled two-dimensional signals," *Proc. IEEE,* vol. 67, pp. 930-949, June 1979.

[9] H. Stark, "Sampling theorems in polar coordinates," *J. Opt. Soc. Am.,* vol. 69, pp. 1519-1525, Nov. 1979.

[10] J. P. Allebach, "Analysis of sampling-pattern dependence in time-sequential sampling of spatiotemporal signals," *J. Opt. Soc. Am.,* vol. 71, pp. 99-105, Jan. 1981.

[11] R. M. Cramblitt and J. P. Allebach, "Analysis of time-sequential sampling with a spatially hexagonal lattice," to appear in *J. Opt. Soc. Am.*

[12] J. P. Allebach, "Design of antialiasing patterns for time-sequential sampling of spatiotemporal signals," to appear in *IEEE Trans. Acoust., Speech, Signal Processing.*

SOME NEW RESULTS ON TREE CODING OF IMAGES

NADER MOAYERI, MORTEZA NARAGHI-POUR, and DAVID L. NEUHOFF
The University of Michigan, ECE Dept.
Ann Arbor, MI 48109

ABSTRACT

This paper compares several different ways of tree coding images. Following the approach of Modestino et.al. [1,2] we employ tree codes based on 2-D DPCM and the (M,L) encoding algorithm. We compare the performance of small and large values of M and L, we compare the traditional raster method of scanning the image with a zigzag approach, and we compare one of the tree codes used in [1,2] with a tree code based on an adaptive version of 2-D DPCM. Our experiments show that although increasing M (the number of paths stored in the search process) improves the performance, such is not the case for the lookahead parameter L. Indeed our results suggest that $L = \log_2(M) + 1$ is the best choice of L for a given M. We found no signficant advantage in zigzag scanning over raster scanning. We observed that the tree code based on adaptive DPCM code improved the signal to noise ratio by about 1 dB. Finally, for the raster scanned approach we have found that parallel implementation of the (M,L)-algorithm using p processors can reduce computation time by a factor of approximately p, provided p is at most $N/(L+1)$, where N is the number of pixels in a horizontal line.

INTRODUCTION

It was first conjectured by Goblick and later proved by Jelinek [3] and Davis and Hellman [4], that tree source codes can perform, in the sense of ensemble average, arbitrarily close to the rate distortion function for memoryless sources. Tan [5] and recently Hashimoto [6] have proved this theorem for stationary ergodic sources. Tree encoding of speech has been extensively studied by Anderson and his co-workers [7,8], where they have evaluated (mostly experimentally) the complexity and performance of such codes.

As for images, two dimensional (2-D) differential pulse code modulation (DPCM) has been the subject of rather extensive studies. Recently Modestino et al. [1,2] have reported the application of 2-D tree coding to synthetic and real world images. They used the (M,L) algorithm [9] to search a tree generated by a 2-D (DPCM) system matched to a 2-D autoregressive image model. We report and compare several variations of the basic encoder employed in [1,2].

In the sections that follow we briefly describe the tree of the code based on non-adaptive 2-D DPCM code, the (M,L) tree encoding algorithm, the two scanning approaches, the adaptive 2-D DPCM approach, the parallel implementation and the experimental results.

THE TREE BASED ON NON-ADAPTIVE 2-D DPCM

An image is represented as an $N \times N$ array $[X_{i,j}]$ of real-valued gray levels. Following Modestino, we have used the decoder of a 2-D DPCM code to generate the tree. See Fig. 1. As in [1,2], the predictor component of the code is given by:

$$Y_{i,j} = \rho_{01}\,\hat{X}_{i,j-1} + \rho_{10}\,\hat{X}_{i-1,j} - \rho_{01}\,\rho_{10}\,\hat{X}_{i-1,j-1}$$

where $\rho_{01} = 0.952$, $\rho_{10} = 0.967$. These coefficients were chosen to equal the corresponding coefficients of a separable autoregressive model tailored to several images. The quantizer has two levels $\pm\Delta$, causing the rate to be 1 bit/pixel. We will refer to Δ as the step-size of the quantizer.

To generate the tree, the DPCM decoder must be fed with all possible $N \times N$ arrays (images) with elements $\pm\Delta$. (It is assumed that $\hat{X}_{0,j} = \hat{X}_{j,0} = 0$ for all j). Each path (codeword) on the tree corresponds to the output of the decoder to one such array. The order in which the elements of the arrays are fed into the decoder, affects the order of the components of the corresponding path in the tree. This will be the subject of the section on scanning.

THE (M,L) ALGORITHM

Following Modestino we have used the (M,L) algorithm as the tree searching scheme. At any given time the algorithm keeps a list of M paths of length L . The cumulative squared error distortion between each path and the image values up to that time is also stored. To proceed one level deeper into the tree the algorithm extends the M paths to $2M$ paths of length $(L+1)$, calculates the new cumulative distortions, finds the subtree containing the best path, and retains up to M best paths from that subtree. It then releases the first bit of that subtree. The main attraction of this algorithm is the synchronized manner in which it produces outputs (no buffer is needed).

SCANNING

When dealing with one dimensional data such as speech there is only one natural order in which to process the samples of the source, namely, the order in which they occur. In two dimensions, however, there is more than one choice of ordering for the processing of the samples. We employed two strategies: A zigzag scanning as well as the more commonly used line by line (raster) scanning. We expected that if neighboring pixels, which are more correlated, are processed without large number of pixels intervening, then better performance would be achieved. This was the motivation for zigzag scanning in which we divided the image into a number of horizontal strips of width W pixels, and encoded the image strip by strip. Fig. 2 shows the ordering in which the pixels of one strip are processed for the case of $W = 4$.

TREE CODING BASED ON ADAPTIVE DPCM

If a tree code is to perform well, there must be sufficient paths in the tree (codewords) to closely follow the typical behavior of images. The capability of the tree generated by 2-D DPCM in this regard is significantly affected by the quantizer step-size Δ. If Δ is small, then the tree paths will not be able to follow a rapid change in the image, such as in an edge. On the other hand, large values of Δ will introduce high distortion where the image is relatively constant. To overcome this dilemma in the choice of Δ we have experimented with a simple adaptive strategy. Depending on present and past values of the sign of $E_{i,j}$, the step-size is changed by a multiplicative factor β. Letting $S_{i,j}$ denote the sign of $E_{i,j}$, the strategy is to choose the step-size $\Delta_{i,j}$ (which is used when encoding $X_{i,j}$) to depend on $\Delta_{i,j-1}$, $S_{i,j-2}$, $S_{i,j-1}$ and $S_{i,j}$. See Table 1. This scheme is a modification of a simple adaptive quantizer used in one dimensional DPCM [10]. Note that it is sensitive only to horizontal variations of the image.

PARALLEL IMPLEMENTATION

In this section we assume line by line scanning. In its usual form the algorithm is sequential, i.e., it processes pixels of the image one row at a time and one pixel at a time. The processing time needed by this scheme is a major stumbling block. However, if small values of L are used in the (M,L) search algorithm (e.g., see Tables 3-6), then it is possible to design a parallel implementation of the encoder which substantially reduces the processing time. The parallel implementation is made possible by the following facts:

1. Suppose the tree has grown to depth (i,N). (The last pixel of the i^{th} row). Corresponding to the last L pixels of the i^{th} row, there are M paths of length L stored in the memory. Due to the fact that the labels $\hat{X}_{i,j}$ for $N-L < j \le N$ do not affect the labels $\hat{X}_{i+1,j}$ for $j \le N-L$, the distortion for pixel at $(i+1,j)$, $j \le N-L$, does not depend on the labels $\hat{X}_{i,j}$ for $N-L < j \le N$. Thus at depth (i,N) the algorithm can choose the path (among M) with the least distortion and release the outputs corresponding to $\hat{X}_{i,N-L+1}, \ldots, \hat{X}_{i,N}$. This is in contrast with the usual operation of the algorithm where at each time it extends the tree one level deeper and releases only the output corresponding to the $(L+1)^{th}$ previous pixel.

2. Suppose the processor is operating on the i^{th} row of the image. There are two branches corresponding to the pixel at $(i,1)$. The labels of these branches are $\hat{X}_{i,1} = \rho_{10}\hat{X}_{i-1,1} \pm \Delta$. One subtree emanates from each branch. After the tree has grown to depth $(i,L+1)$, the algorithm chooses one of the two subtrees and releases the corresponding output bit. Thus at the depth of $(i,L+1)$ the label $\hat{X}_{i,1}$ is either $\rho_{10}\hat{X}_{i-1,1} + \Delta$ or $\rho_{10}\hat{X}_{i-1,1} - \Delta$, and therefore, the labels $\hat{X}_{i+1,1}$ can be computed since

$$\hat{X}_{i+1,1} = \rho_{10}\hat{X}_{i,1} \pm \Delta .$$

Facts 1 and 2 imply that a number of parallel processors can be used to encode the image. Each processor will be operating on one row of the image and with a delay of $(L+1)$ pixels with respect to the previous processor (operating on the previous row). At most $\left\lceil \frac{N}{L+1} \right\rceil$ processors can be operating at any given time. (If more processors are used, some of them will be left idle.) It can be easily verified that the processing time reduces by the number of processors p, if $p \le \left\lceil \frac{N}{L+1} \right\rceil$. For $p > \left\lceil \frac{N}{L+1} \right\rceil$, the reduction factor is still $\left\lceil \frac{N}{L+1} \right\rceil$.

RESULTS

The encoding techniques were tested on an outdoor closeup scene of a girl next to a tree. Results are tabulated in Tables 2-6. Table 2 shows the signal to noise ratio of a 2-D DPCM code for different values of the step-size Δ. (Signal to noise ratio is the ratio of empirical signal variance to empirical noise variance.) Signal to noise ratio for tree coding is tabulated in Tables 3-6 along with the CPU time of a VAX 11/780 machine. In Table 3, line by line scanning is used for different values of M and L and a fixed step-size Δ. The results of this table suggest the following: while increasing M improves the performance (although negligibly for $M = 16$ & 32), such is not the case for L. Smaller values of L may result in better performance and $L = 1 + \log_2 M$ seems to be the best choice. This might be due to the merging of paths which will effectively reduce the gain due to the multi-path feature of tree coding. In Table 4 we show the results for zigzag scanning, for different values of M, L and strip width W. ($W = 1$ corresponds to line by line scanning.) There are only very slight improvements over the results for line by line scanning. Adaptive tree coding was tried next and the results are given in Tables 5 and 6. In Table 5, different values of Δ and β (see Table 1) are tried for adaptive DPCM and tree coding with $M = 4$ and $L = 2$. The values $\Delta = 6, \beta = 1.3$ and $\Delta = 7, \beta = 1.2$ seem to achieve the highest signal to noise ratio. (We presume, of course, that these best values are dependent on the image.) In Table 6, M and L are changed for $\Delta = 6$, $\beta = 1.3$. The same conclusions can be drawn about M and L as for the non-adaptive case.

CONCLUSIONS

The following conclusions can be drawn from our results:

1. Zigzag scanning increases the computational complexity of the encoder and contrary to our expectation, no significant improvement in performance was obtained.

2. Performance improves with M. Large values of M e.g., (32-64) are prohibitive due to the complexity of the encoder and $M = 4$ or 8 is a good compromise.

3. Large values of L are unnecessary and may even degrade the performance. Our results suggest that for a given M, $L = 1 + \log_2 M$ is the best choice.

4. A simple adaptive scheme can improve the performance by about 1 dB.

5. According to our observations, tree coding is about 2 dB better than DPCM. The corresponding figure for adaptive tree coding and adaptive DPCM is about 2.6 dB. Of course these figures are only representative and will vary somewhat from image to image.

REFERENCES

(1) J.W. Modestino, V. Bhaskaran and J.B. Anderson, "Tree Encoding of Images in the Presence of Channel Errors." IEEE Trans. on Information Theory, vol. IT-27, pp 677-697, Nov. 1981.

(2) J.W. Modestino, V. Bhaskaran, "Robust Tow-Dimensional Tree Encoding of Images." IEEE Trans. on Communications, vol. COM-29, pp 1786-1798, Dec. 1981.

(3) F. Jelinek, "Tree Encoding of Memoryless Time-Discrete Sources with a Fidelity Criterion." IEEE Trans. on Information Theory, vol. IT-15, pp 584-590, Sept. 1969.

(4) C.R. Davis and M.E. Hellman, "On Tree Coding with a Fidelity Criterion." IEEE Trans. on Information Theory, vol. IT-21, pp 373-377, July 1975.

(5) H. Tan, "Tree Coding of Discrete-Time Abstract Alphabet Stationary Block-Ergodic Sources with a Fidelity Criterion." IEEE Trans. on Information Theory, vol. IT-22, pp 671-681, Nov. 1976.

(6) T. Hashimoto, "Tree Encoding of Stationary Ergodic Source with a Fidelity Criterion." IEEE Trans. on Information Theory, vol. IT-29, pp 650-656, Sept. 1983.

(7) J.B. Anderson and J.B. Bodie, "Tree Encoding of speech." IEEE Trans. on Information Theory, vol. IT-21, pp 379-387, July 1975.

(8) S. Mohan and J.B. Anderson, "Speech Encoding by the Stack Algorith." CRL Internal Report Series, No. CRL-64, McMaster University, May 1979.

(9) F. Jelinek and J.B. Anderson, "Instrumentable Tree Encoding of Information Sources." IEEE Trans. on Information Theory, vol. IT-17, pp 118-119, Jan. 1971.

(10) L.H. Goldstein and B. Liu, "Deterministic and Stochastic Stability of Adaptive Differential Pulse Code Modulation." IEEE Trans. on Information Theory, vol. IT-23, pp 445-453, July 1977.

$S_{i-2,j}$	$S_{i-1,j}$	$S_{i,j}$	$\Delta_{i,j}$
-1	-1	-1	$\beta\,\Delta_{i-1,j}$
-1	-1	+1	Δ
-1	+1	-1	$-\Delta_{i-1,j}/\beta$
-1	+1	+1	$\beta\Delta$
+1	-1	-1	$-\beta\Delta$
+1	-1	+1	$-\Delta_{i-1,j}/\beta$
+1	+1	-1	$-\Delta$
+1	+1	+1	$\beta\Delta_{i-1,j}$

Table 1 Strategy for changing the step-size in adaptive schemes

Δ	SNR (in dB)
9	16.40
10	16.44
11	16.44
12	16.49
13	16.20
14	15.90

Table 2 Results for 2-D DPCM

M	L	SNR (in dB)	CPU Time (in seconds)
2	1	17.79	198
	2	17.90	232
	3	17.76	260
	4	17.72	268
	5	17.66	267
	6	17.62	267
	128	17.13	1070
	256	17.02	1360
4	2	18.22	390
	3	18.24	442
	4	18.13	483
	5	18.16	538
	6	18.13	538
	7	18.09	557
	128	17.48	2084
	256	17.49	2498
	512	17.30	NA
8	3	18.30	770
	4	18.28	884
	5	18.31	976
	16	18.11	NA
	128	17.74	3921
	256	17.62	5046
	512	17.67	NA
16	4	18.39	NA
	5	18.42	NA
	8	18.11	NA
32	5	18.49	NA

Table 3 Results for tree coding with $\Delta = 10$ (non-adaptive, raster scanned) NA = Not Available

M	L	W	SNR (in dB)	CPU Time (in seconds)
4	2	2	18.04*	439
	2	4	17.73	392
	3	2	18.04*	443
	4	2	18.22*	447
	4	4	18.08*	477
	8	1	18.08	NA
	8	4	18.21	530
8	8	1	18.30	NA
	8	4	18.37	1025
32	8	4	18.58	NA
	12	4	18.63	NA
	16	4	18.60	NA
	16	8	18.56	NA

Table 4 Results for tree coding with zigzag scanning and $\Delta = 10$ (non-adaptive) NA : Not Available

* : Compare with table 3

Δ	β	Tree Coding		ADPCM (in dB)
		SNR (in dB)	CPU Time (in seconds)	SNR
10	1.2	18.16	429	15.12
9	1.3	18.31	435	13.18
8	1.1	18.88	391	15.88
	1.2	18.90	435	14.77
	1.3	18.70	404	13.84
7	1.1	18.72	423	15.25
	1.2	19.10	437	14.12
	1.3	18.90	406	12.88
6	1.2	18.99	411	13.58
	1.3	19.10	407	11.15
	1.4	18.98	397	
5	1.3	18.80	438	9.17
	1.4	19.06	373	
	1.4	18.92	433	

Table 5 Results for ADPCM and adaptive tree coding with $M = 4$, $L = 2$ (raster scanned)

M	L	SNR (in dB)	CPU Time (in seconds)
4	2	19.10	407
	3	19.20	440
	4	18.97	503
	5	19.19	501
	8	18.99	527
8	3	19.37	798
	4	19.47	943
	5	19.47	990
16	4	19.57	1781
	5	19.63	1992
	6	19.57	2207
32	5	19.70	3205
	6	19.72	4204
	7	19.70	4597

Table 6 Results for adaptive tree coding with $\Delta = 6$, $\beta = 1.3$ (raster scanned)

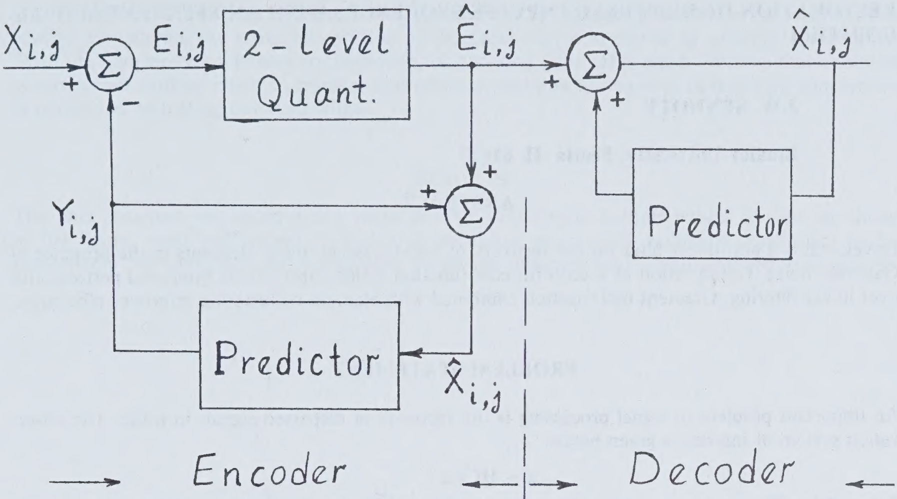

$X_{i,j}$
$E_{i,j}$
2-Level Quant.
$\hat{E}_{i,j}$
$\hat{X}_{i,j}$
Σ
+
−
$Y_{i,j}$
Predictor
Predictor
$\hat{X}_{i,j}$
Encoder
Decoder

Fig.1_ 2-D DPCM Code

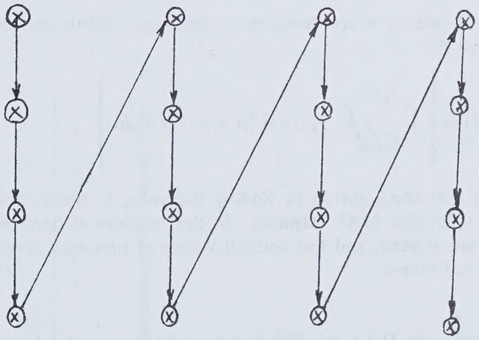


Fig. 2_ Zig zag scanning for W = 4.

RESTORATION OF DISPERSED IMPULSE SEQUENCES WITH AN EFFICIENT MAP ALGORITHM

J.W. SENNOTT

Bradley University, Peoria IL 61625

ABSTRACT

Developed is a non-linear filter for the recovery of non-Gaussian image elements in the presence of Gaussian noise. Optimization of a uniform cost function yields substantially improved performance over linear filtering. Gradient optimization combined with element reclustering improves efficiency.

PROBLEM STATEMENT

An important problem in signal processing is the recovery of dispersed signals in noise. The observation system of interest is given below:

$$\mathbf{g} = \mathbf{Hf} + \mathbf{n}$$

For additive Gaussian noise the maximum- likelihood solution is the inverse filter, which in the discrete case is given by:

$$\hat{\mathbf{f}} = \left\{ \mathbf{H}'\,\mathbf{H} \right\}^{-1} \mathbf{H}'\,\mathbf{g}$$

For signal vectors **f** of mixed distribution the inverse filter yields particularly poor performance. In this paper densities are impulsive near the origin and smooth for positive values. A risk function that appears tractable is the uniform function:

$$R = Pr\left[\bigcup_{i=1}^{N} \{ \, | \hat{f}_i - f_i \, | > \Delta \, \} \right]$$

which is the probability of one or more component estimates falling outside a tolerance Δ. The functional to be maximized is:

$$\underset{\hat{f}_i}{Max} \left\{ \int_{\hat{f}_i - \Delta}^{\hat{f}_i + \Delta} \cdots \int p_{\mathbf{g}|\mathbf{f}}\,(\,\beta\,|\,\alpha\,)\; p_{\mathbf{f}}\,(\,\alpha\,)\; d\alpha \right\}$$

For smooth densities one gets the solution by finding the value of the vector α for which the integrand is maximum, the familiar MAP estimate. In the problem at hand we must jointly determine which components are at zero, and the optimal values of non-zero components. To this end we rewrite the functional as follows:

$$\underset{ZS}{Max}\; \underset{\hat{\mathbf{f}}_1}{Max} \left\{ -(\mathbf{g} - \mathbf{H}\,\hat{\mathbf{f}}_1)'\,(\mathbf{g} - \mathbf{H}\,\hat{\mathbf{f}}_1\,) + (N - NR)\,\ln\,(1-k) + \sum_{i=0}^{NR} \ln\, p_{f_i} \right\}$$

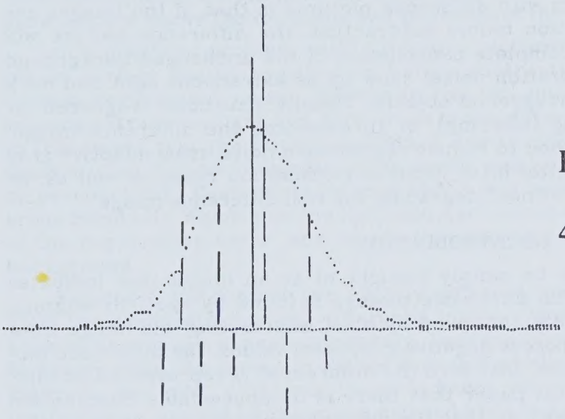
This at first appears to be a burdensome problem. Now, for densities of interest it is reasonable to assume that:

$$(1 - k) >> p_{f_i}\;, \quad \hat{f}_i > 0$$

Observe that under this assumption the functional is only increased by increasing the number of elements in the zero set, provided that the terms involving (g- Hf) are changed by only a small amount. This condition is met by the reclustering method which we propose. Multiple non-zero components falling within a resolution window are combined if no other window is found with a greater number of non-zero elements. Ties are resolved by computing a posteriori probability. Following reclustering the remaining portion of the functional is increased by gradient optimization. Once the non-zero components are improved, reclustering again takes place. Observe that clustering is carried out without resort to gradient optimization, and that the number of non-zero components is reduced at each stage in the algorithm.

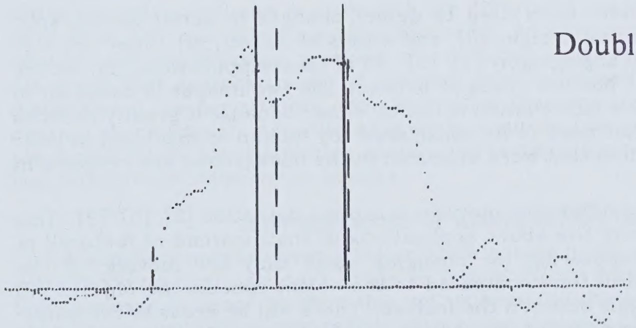
RESULTS

The filter structure was tested over a range of SNR values with multiple pulses. Results are shown in the figure, solid and dashed vertical lines indicating correct and estimated component values respectively. An advantage of approximately 20 dB was observed over a well designed inverse filter.



Inverse Filter Result

40 dB SNR



Double Pulse 20 dB SNR

ADAPTIVE GRAY SCALE MAPPING TO REMOVE REGISTRATION NOISE IN DIFFERENCE PICTURES*

THOMAS F. KNOLL and EDWARD J. DELP
Computer, Information, and Control Engineering
and
Department of Electrical and Computer Engineering
University of Michigan
Ann Arbor, Michigan

ABSTRACT

Difference pictures are used in various image processing applications such as change detection, radar imaging, remote sensing, and biomedical image analysis. The difference picture, or difference image, is found by subtracting one image from another. One practical problem with difference pictures is that, if the images are not in perfect spatial registration before subtraction, the difference picture will contain artifacts caused by incomplete cancellation of the unchanged background objects. These artifacts (registration noise) show up as extraneous light and dark regions on either side of the background objects. Usually, this noise is ignored, or is reduced by either smoothing (blurring), or thresholding the difference image. This paper describes a new method to reduce registration noise using adaptive gray scale mapping. This simple digital filter reduces registration noise as well as, or better than, previous methods, without degrading the real difference image.

1.0 INTRODUCTION

The difference picture can be simply thought of as an image that indicates change between two images. The difference picture is found by spatially aligning the two images as well as possible and, pixel by pixel, subtracting one image from the other. To avoid having to process negative gray level values, the difference picture gray levels are often offset so that zero (no difference) is represented as middle gray. We shall assume in this paper that there is no appreciable illumination difference between the two images, so that the difference picture can be used as a measure of what is actually different between the two scenes. In many applications one would have to adjust for illumination effects before the difference picture could be found.

Difference pictures have been used to detect changes in aerial photographs [9], radar images [6], Landsat images [8], and x-rays of bones [10]. They are also used in digital subtraction angiography [1], [5]. All of these applications rely on the fact that any feature that has not changed between the two images is canceled in the difference picture. This cancellation is useful, either because it greatly reduces the number of features that need to be considered (by human or machine) in later analysis, or because features that were obscured in the background are revealed in the difference picture.

Another application of difference pictures is motion detection [2], [3], [7]. This application is different from the above applications in that, instead of features or objects appearing or disappearing (or changing size), they are moving. If the frames are closely spaced in time compared to the object speeds, the objects will have moved only a few pixels between the frames. There will be areas in the difference picture at the leading edges of the objects, caused by the objects covering the background, and areas at the trailing edges, caused by the background being uncovered. These areas present a problem, as they look exactly like registration noise. (Registration noise is also caused by a displacement between the two frames—it is as if the entire background is moving). It is impossible for any local

---

*This research was partially sponsored by the Biomedical Research Council and the Office of the Vice-President for Research, University of Michigan.

operation to remove registration noise from these types of difference pictures without destroying the motion information. This situation will not be considered in this paper.

However, if the frames are widely spaced in time compared to the object speeds, the objects will have moved a large distance (several object diameters) between the frames. This can be thought of as simple change detection; objects have disappeared from some locations, and other objects have appeared at different locations. This paper will consider the removal of registration noise from these difference pictures.

We have used difference pictures to detect molecules in tissue samples [4]. Two consecutive slices are used as samples, so they are similar in shape and features. One sample is digested with an enzyme that removes a specific molecule, while the other sample is untreated. Areas that contains the molecule will be darker in the control sample than in the treated sample, whereas areas that do not contain the molecule will have the same gray level in both samples. Thus the difference picture shows the locations and densities of the removed molecule.

One practical problem with difference pictures is that, if the two images are not in perfect spatial registration before subtraction, the difference picture will contain registration noise. Objects in the two images will not line up perfectly, causing extraneous light and dark areas in the difference picture. (We shall assume in this paper that the misregistration is due to changes in camera angle or small translational motion of objects in the scene, and not to changes in camera distance or perspective). Each misregistered object will have corresponding light and dark areas beside it. Which side the light and dark areas are on depends on the direction of the registration error, and on whether the object is lighter or darker than the background.

Figures 1 and 2 show two slightly different images, and Figure 3 shows their ideal (no registration noise) difference image. Notice how the objects around the edge of the image cancel completely, because they are identical in both images. However, when the image in Figure 2 is shifted slightly before subtracting it from Figure 1, the "raw" difference image is found (shown in Figure 4). Notice the registration noise, caused by the misregistered objects around the edge of the image.

Even a slight misregistration will add noise to the difference image, possibly overwhelming the real differences between the images. Removing this noise will assist human or machine interpretation of the difference images. One method to reduce registration noise is to smooth the difference image; this has the side effect of blurring the real differences (see Figure 5). Another common method is to threshold the difference image, mapping all pixels in a range around middle gray to middle gray. This destroys much of the gray scale information in the picture, and is ineffective if the registration error is greater than one pixel (assuming the images contain sharp edges). Adaptive gray scale mapping (discussed later) is a new, simple digital filter that is effective in reducing registration noise, while preserving the real differences between the images.

2.0 PROPERTIES OF REGISTRATION NOISE

A simple property of registration noise is that it has a zero sample mean. The corresponding light and dark areas on either side of each background object spatially average to zero. To show this, let us compute the spatial average of a difference image containing only registration noise. Let $g(i,j)$ be the difference image between a discrete image $f(i,j)$, containing objects on a constant background, and a shifted version of that same image.

$$g(i,j) = f(i,j) - f(i-\Delta i, j-\Delta j)$$

This difference image will consist totally of registration noise. Let $f(i,j)$ be equal to a constant c, outside some region $A \in I^2$ (I is the set of all integers). Let region $B \in I^2$ be the region A shifted by $(\Delta i, \Delta j)$. Summing the difference (noise) image over all non-zero values gives

$$\sum\sum_{(i,j)\in A\cup B} g(i,j) = \sum\sum_{(i,j)\in A\cup B} f(i,j) - f(i-\Delta i, j-\Delta j)$$

$$= \sum\sum_{(i,j)\in A\cup B} f(i,j) - \sum\sum_{(i,j)\in A\cup B} f(i-\Delta i, j-\Delta j)$$

$$= \sum\sum_{(i,j)\in A} f(i,j) - \sum\sum_{(i,j)\in B} f(i-\Delta i, j-\Delta j) + \sum\sum_{(i,j)\in \bar{A}\cap B} c - \sum\sum_{(i,j)\in A\cap \bar{B}} c$$

and using the fact that regions $\bar{A}\cap B$ and $B\cap\bar{A}$ have the same area

$$= \sum\sum_{(i,j)\in A} f(i,j) - \sum\sum_{(i,j)\in B} f(i-\Delta i, j-\Delta j)$$

$$= \sum\sum_{(i,j)\in A} f(i,j) - \sum\sum_{(i,j)\in A} f(i,j)$$

$$= 0$$

Thus, the registration noise caused by each object or set of objects will spatially average to zero.

The above proof assumes that the images can be modeled as objects on a constant background. (Non-uniform gray level objects are allowed—they are just treated as multiple, overlapping uniform objects). An image which could not be modeled as objects on a background is an infinite step reference scale. Here the registration noise is monochromatic (all light or all dark). Fortunately, most images can be approximately modeled as objects on a background.

In general, corresponding light and dark regions can be far apart, separated by as much as the largest object dimension. A large averaging neighborhood would be required for the registration noise to average out. The noise reduction problem is simplified if most of the corresponding light and dark areas of registration noise are close together. Then only a small neighborhood is required for the noise to spatially average to zero. Registration noise from small objects is close together, as is noise from long thin objects (if the end effects are ignored). For many images the approximation that corresponding regions are near each other is a good one, because most images contain many more small objects than large objects. Small objects cause more noise per area of the object than large objects (a long thin object will cause just as much noise as a fat object of equal length).

Both smoothing and adaptive gray scale mapping will use the fact that registration noise is zero mean, and will assume that corresponding light and dark regions are near each other.

3.0 CURRENT METHODS USED TO REMOVE REGISTRATION NOISE

The best way to reduce registration noise is to register the images accurately before subtracting. Most of the work in the reduction of registration noise has addressed the problem by proposing better registration algorithms [6], [9]. Both linear and higher order geometrical transforms are used to align the images as accurately as possible. However, to completely prevent registration noise, the images must be aligned to sub-pixel accuracy. Even if the images are registered carefully beforehand, some noise is likely to remain in the difference picture.

Sometimes it is not possible to align the images accurately enough before subtraction. One case is subtractive television [7]. Here the subtraction is done by the camera; the only input to the computer is the difference image. Registration error caused by non-repeatability of the camera cannot be corrected beforehand, and must be removed from the difference image.

Another case where it is not possible to align the images accurately enough is our work with consecutive tissue samples [4]. Although the samples are very similar, they are not identical. While the basic outline of the samples can be aligned, small scale features, such as cells, usually cannot. Again, the noise must be removed after computing the difference image.

3.1 Smoothing to Remove Registration Noise

The simplest way to remove registration noise is to smooth the image by replacing each pixel by the average gray level of pixels in its neighborhood. This can be either an unweighted (simple) or a weighted average. If the averaging neighborhood is large enough, it will probably contain both of the corresponding light and dark areas of registration noise. Because the noise is zero mean, it will average out and be removed.

The problem with smoothing is that, if the averaging neighborhood is large enough to reduce the registration noise, it also blurs the real differences between the images. Figure 5 shows the the raw difference image in Figure 4 after simple smoothing. The registration noise is greatly reduced, but the real differences have been so blurred that they are hardly visible.

3.2 Thresholding to Remove Registration Noise

If the registration error is less than one pixel (slightly more if the images contain no sharp edges), thresholding can be used to reduce registration noise. When the registration error is small, the registration noise is likely to be less intense (closer to middle gray) than the real difference image. By setting pixels whose gray level is within a certain distance of middle gray to middle gray, noise can be reduced. Other pixels can either be left alone, or set to the corresponding maximum or minimum value.

One problem with thresholding is that it fails if the registration error is larger than one pixel. Registration noise caused by a sharp edge and an alignment error of only one pixel will be just as or more intense as the real differences between the images. If the image in Figure 4 is thresholded, it noise is totally uneffected.

4.0 ADAPTIVE GRAY SCALE MAPPING TO REMOVE REGISTRATION NOISE

Adaptive gray scale mapping is a filter that, for each pixel, uses its neighborhood to estimate whether the pixel is part of the true difference image, or part of the registration noise. A non-linear gray scale map is computed for each pixel, based on the amount of "brightness" and "darkness" in a neighborhood around that pixel. Each pixel is then passed through its own map to produce the filtered difference image.

Adaptive gray scale mapping can work much better than simple smoothing. Figure 6 shows Figure 4 after adaptive gray scale mapping. The noise is reduced just as much as in Figure 5, but the real difference image is not blurred.

Imagine the difference image as a sandbox. Where the difference image is light, there is a hill; where it is dark, there is a valley. Registration noise shows up as corresponding hills and valleys. Because registration noise has a zero sample mean, each hill contains exactly as much sand as is missing from its corresponding valley. In effect, all the registration noise does is scoop up sand from some parts of the image, leaving valleys (dark areas), and dump it in other parts, creating hills (light areas). Adaptive gray scale mapping tries to reverse this process.

To remove registration noise, the sand must be removed from the hills and put back in the valleys. One way to do this is to remove sand from all the hills (in proportion to their height), and dump it into all the valleys (in proportion to their depth). This continues until either there are no more hills, or there are no more valleys. The problem with this method is that all hills and all valleys are treated alike; isolated hills (probably part of the real difference image) are chopped just as much as are hills next to valleys (which are probably registration noise).

The algorithm is refined by using the assumption that corresponding light and dark areas (hills and valleys) caused by registration noise are near one another. Sand from a hill is used to fill only the nearby valleys. Isolated hills or valleys are not effected, whereas areas that contain both hills and valleys that are near each other are smoothed out. The final image will now contain some areas with only hills, some areas with only valleys, but no areas with both hills and valleys.

This algorithm cannot be implemented exactly as described. The sand moving process is parallel and continuous. Sand is moved from all the hills to all the valleys at the same time. For a given hill and valley, the rate of sand transfer between them will depend inversely on their distance, and their *current* (not *initial*) height and depth. The algorithm would have to be approximated by many iterations, each moving a small amount of sand.

The algorithm is speeded by, instead of actually moving the "brightness" (sand) around, the final gray level of a pixel is estimated from the amount of "brightness" and "darkness" in the neighborhood around that pixel. For example, if a pixel is below the reference level (in a valley), and there is equal brightness and darkness (sand in hills and sand missing from valleys) in its neighborhood, the pixel is mapped to the reference level (the valley is filled in). Or, if there is twice as much darkness as brightness, the pixel is mapped to a gray level halfway between its current level and the reference level (the valley is halfway filled in). A non-linear gray scale map is computed for each neighborhood, based of the amount of brightness and darkness in it, and its center pixel is then mapped to a new value.

4.1 Implementation of Adaptive Gray Scale Mapping

Adaptive gray scale mapping has two parameters. The first is the radius Δ, of the square neighborhood around each pixel that sand could be moved to or from. If the neighborhood is too small, it will not contain both the light and dark regions of registration noise, and the algorithm will not remove the noise. However, for an object in the real difference image is not to be effected by the filter, it must be isolated from regions of opposite color by at least the neighborhood radius. As the radius is increased, more and more of the real difference image will not be isolated enough, and will be mistakenly dimmed or completely removed. The ideal radius is large enough to reduce noise, but small enough to not excessively dim the real difference image. It must, in general, be found by trial and error for each class of difference images.

The second parameter is the reference gray level r, used to separate hills and valleys. A pixel brighter than the reference is considered to be part of a hill; a pixel darker is considered to be part of a valley. Normally, the reference level is set to the gray level in the difference picture that corresponds to no difference between the two input images. Areas that contain only registration noise will contain equal moments above and below this reference level. Because both moments are equal, the noise will be completely removed. If the reference level is set to some other level, the moments will not be equal, and the noise will be only partially removed.

However, if the background gray level has changed between the two images, a different reference level will achieve greater noise reduction. Registration noise in these areas will not average to zero, but instead to the difference in the background levels. Maximum noise reduction is then achieved when the reference level is set to the difference in background levels.

Two conditional local moments are computed for each pixel in the difference image. The first is the brightness moment

$$b(i,j) = \sum_{m=i-\Delta}^{i+\Delta} \sum_{n=j-\Delta}^{j+\Delta} (f(m,n)-r)\, \mathrm{I}\{f(m,n)>r\}$$

where I is the indicator function. Pixels outside the image are assumed to be at the reference level. In the sand model, the brightness moment corresponds to the amount of sand in the hills near the pixel. The other moment computed is the darkness moment

$$d(i,j) = \sum_{m=i-\Delta}^{i+\Delta} \sum_{n=j-\Delta}^{j+\Delta} (r - f(m,n))\, \mathrm{I}\{ f(m,n) < r \}$$

This corresponds to the amount of negative sand, or air, in the valleys near the pixel.

The gray level of each pixel in the difference image is then mapped to a new gray level

$$g(i,j) = \begin{cases} r + (f(i,j) - r)\dfrac{b(i,j) - d(i,j)}{b(i,j)} & \text{if } f(i,j) > r \text{ and } b(i,j) > d(i,j) \\ r - (r - f(i,j))\dfrac{d(i,j) - b(i,j)}{d(i,j)} & \text{if } f(i,j) < r \text{ and } b(i,j) < d(i,j) \\ r & \text{otherwise} \end{cases}$$

If a pixel is in the minority, i.e. a dark pixel in a mostly bright area, or a bright pixel in a mostly dark area, it is mapped to the reference level. If the pixel is in the majority, its contrast is toned down with a ramp function whose slope depends on the amount of disagreement in the area. For example, if there is three times as much dark as bright, the ramp will have slope two-thirds. (In the sandbox model, this corresponds to filling the valley one-third of the way).

Adaptive gray scale mapping can be done quickly. If the conditional moment calculations are implemented as written, execution time would be proportional to the image area times the neighborhood area. However, by using incremental updating to compute the moments, the execution time is reduced so that it is proportional to only the image area. This is done by first computing column subtotals, and then summing the subtotals. The moments are incrementally updated as the row is scanned, by adding in the leading subtotals, and subtracting out the trailing subtotals. When changing to the next row, each of the column subtotals is incrementally updated by looking at the leading and trailing rows of pixels.

4.2 Preservation of Neighborhood Mean Gray Level

Because registration noise has a zero mean, it is desirable that a filter that removes registration noise should not change the mean gray level of the difference picture. We shall show in this section that adaptive gray scale mapping approximately preserves the neighborhood mean gray level.

Computing the neighborhood sum of the difference picture around (i,j), assuming $b(i,j) \geq d(i,j)$

$$\sum_{m=i-\Delta}^{i+\Delta} \sum_{n=j-\Delta}^{j+\Delta} g(m,n) =$$

$$\sum_{m=i-\Delta}^{i+\Delta} \sum_{n=j-\Delta}^{j+\Delta} \left[r + \frac{b(m,n) - d(m,n)}{b(m,n)} (f(m,n) - r)\, \mathrm{I}\{ f(m,n) > r \} \right]$$

Assuming that neighborhoods of pixels in the neighborhood of (i,j) all have about the same amount of brightness and darkness, i.e.

$$b(m,n) \approx b(i,j)$$

$$d(m,n) \approx d(i,j)$$

This approximation holds well if the neighborhood size is large compared to the characteristic dimension of the texture field. Using this approximation

$$\sum_{m=i-\Delta}^{i+\Delta} \sum_{n=j-\Delta}^{j+\Delta} g(m,n) \approx \left[\sum_{m=i-\Delta}^{i+\Delta} \sum_{n=j-\Delta}^{j+\Delta} r \right] + \frac{b(i,j) - d(i,j)}{b(i,j)} \times$$

$$\sum_{m=i-\Delta}^{i+\Delta} \sum_{n=j-\Delta}^{j+\Delta} (f(m,n) - r)\, \mathrm{I}\{ f(m,n) > r \}$$

$$= \left[\sum_{m=i-\Delta}^{i+\Delta} \sum_{n=j-\Delta}^{j+\Delta} r \right] + \frac{b(i,j)-d(i,j)}{b(i,j)} \, b(i,j)$$

$$= \sum_{m=i-\Delta}^{i+\Delta} \sum_{n=j-\Delta}^{j+\Delta} r + \sum_{m=i-\Delta}^{i+\Delta} \sum_{n=j-\Delta}^{j+\Delta} (f(m,n)-r)$$

$$= \sum_{m=i-\Delta}^{i+\Delta} \sum_{n=j-\Delta}^{j+\Delta} f(m,n)$$

This can also be shown when $b(i,j) \leq d(i,j)$. Thus the neighborhood sum, (and thus the neighborhood mean), are approximately preserved by adaptive gray scale mapping.

5.0 EXAMPLE: MOLECULE DETECTION USING CONSECUTIVE TISSUE SAMPLES

We have used difference pictures to detect the locations and densities of specific molecules in tissue samples [4]. Two consecutive tissue slices are used. One is kept as a control, while the other is treated with an enzyme that digests the molecule in question. Both samples are stained and photographed. The photographs are then digitized and registered as well as possible. The treated image is then subtracted from the control image. The resulting difference picture shows, in theory, the locations and densities of the molecule in question.

The problem with this method is that the two samples are not identical. While the basic outline of the samples can be aligned, the smaller scale features, such as cells, cannot. This is because the cell dimensions are of the same order as the slice thickness. Although each sample will have about the same density of cells, the cells will not line up between the two images. The unfiltered (raw) difference picture will contain of lot of registration noise.

Figures 7 and 8 show the digested and the control images before subtraction. Figure 9 shows their raw difference image. Notice how the registration noise (extraneous light and dark regions) makes it difficult to see which areas are really darker in the control image.

The raw difference image was then filtered using adaptive gray scale mapping. The reference gray level was set to the value corresponding to no difference between the input images. The region radius was found be trial and error—starting with a small value and increasing it in steps until enough noise reduction was obtained. The resulting filtered difference image is shown in Figure 10. Notice how it is much easier to see where, and how much, the control sample is darker than the treated sample (and thus the locations and densities of the molecule in question).

In this application, the texture of the tissue areas in the difference images is not important, only their mean gray levels. The texture is an artifact caused by the cells, and should be ignored. This texture could be removed by smoothing the difference image, but that would blur details in other areas of the difference image.

6.0 CONCLUSIONS

Difference pictures are useful in detecting change between two images. One problem with difference pictures is that, if the two images are not aligned perfectly before subtraction, extraneous light and dark regions will appear in the difference image. These artifacts are caused by the misaligned background objects in the two images not canceling completely in the difference picture. We have presented a new algorithm using adaptive gray scale mapping, that reduces registration noise from difference pictures. It removes noise as well as or better than existing methods, while not blurring the real differences between the images. It also preserves the mean gray level, which is desirable because registration noise does not change the mean gray level.

We would like to thank Dr. Linda L. Brinkley of the Department of Anatomy and Cell Biology, University for Michigan, for suggesting the above problem and providing technical support and advice.

7.0 REFERENCES

[1] C. K. Chow, S. K. Hilal, K. E. Niebuhr: X-ray Image Subtraction by Digital Means, *IBM Journal of Research and Development* Vol. 17, 1973, pp. 206-218

[2] R. Jain: Extraction of Motion Information from Peripheral Processes, *IEEE Transactions of Pattern Analysis and Machine Intelligence* Vol. PAMI-3, 1981, pp. 489-503

[3] R. Jain: Dynamic Scene Analysis Using Pixel-Based Processes, *Computer* Vol. 14, August 1981, pp. 12-18

[4] T. F. Knoll, L. L. Brinkley, E. J. Delp: Analysis of Histological Images Using Difference Pictures, Submitted to *Analytical and Quantitative Cytology*

[5] R. S. Ledley, L. M. Arminski, L. S. Rotolo, T. J. Golab, C. M. Park: TEXAC-Based Digital Fluoroscopy, *IEEE Computer Society Conference on Pattern Recognition and Image Processing*, 1982, pp. 530-533

[6] R. L. Lillestrand: Techniques for Change Detection, *IEEE Transactions on Computers* Vol. C-21, 1972, pp. 654-659

[7] M. Onoe, N. Hamano, K. Ohba: Computer Analysis of Traffic Flow Observed by Subtractive Television, *Computer Graphics and Image Processing* Vol. 2, 1973, pp. 377-392

[8] C. J. Robinove, P. S. Chavez, D. Gehring, R. Holmgren: Arid Land Monitoring Using Landsat Albedo Difference Images, *Remote Sensing of Environment* Vol. 11, 1981, pp. 133-156

[9] M. S. Ulstad: An Algorithm for Estimating Small Scale Differences Between Two Digital Images, *Pattern Recognition* Vol. 5, 1973, pp. 323-333

[10] R. L. Webber, U. E. Ruttimann: Digital Subtraction of Radiographs as an Aid for Detecting Pathologic Changes in Periodontal Bone, *IEEE 1982 International Workshop on Physics and Engineering in Medical Imaging*, pp. 130-136

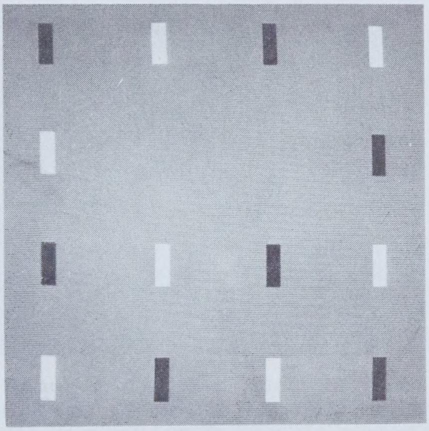


Figure 1: A 256x256 pixel test image that contains both dark and light objects on a constant background.

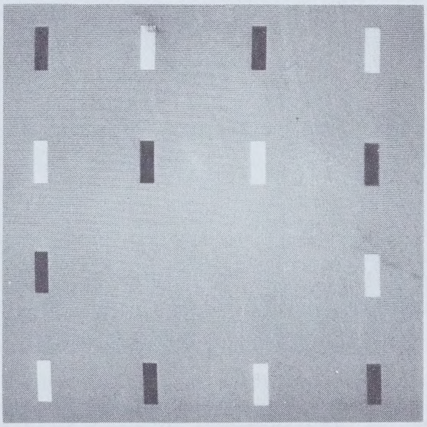


Figure 2: A second test image, almost the same as Figure 1, except for the center objects.

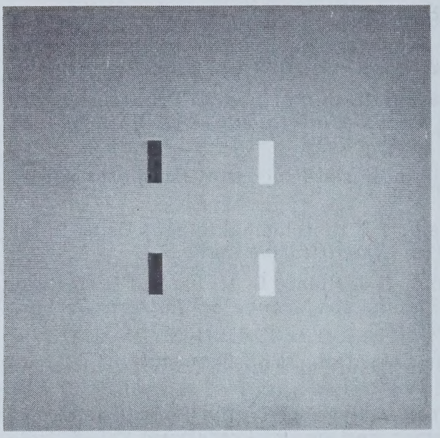


Figure 3: The ideal difference image found by subtracting Figure 2 from Figure 1. Note how the objects around the edge completely cancel out since they are the same in both images.

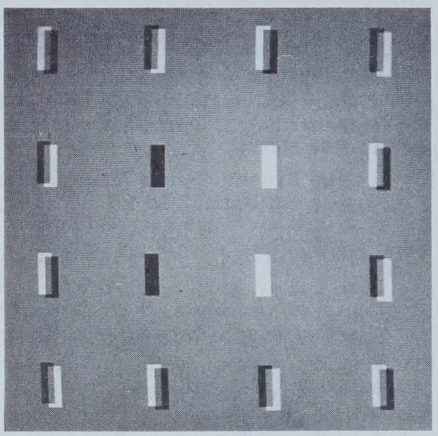


Figure 4: The raw difference image between Figure 1 and a shifted version of Figure 2. Figure 2 has been shifted by 4 pixels in the horizontal direction and 2 pixels in the vertical.

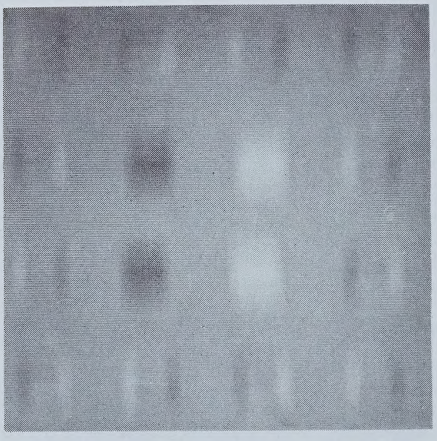


Figure 5: The raw difference image from Figure 4 after simple smoothing with a neighborhood of 25 by 25 pixels. The registration noise is reduced, but the difference image is blurred.

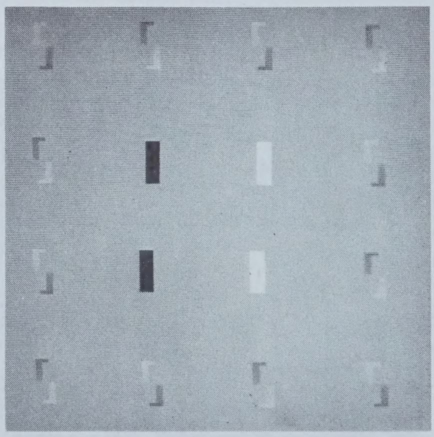


Figure 6: The raw difference image from Figure 4 after adaptive gray level mapping with a neighborhood of 25 by 25 pixels. The registration noise is almost completely removed, and the difference image is left unchanged.

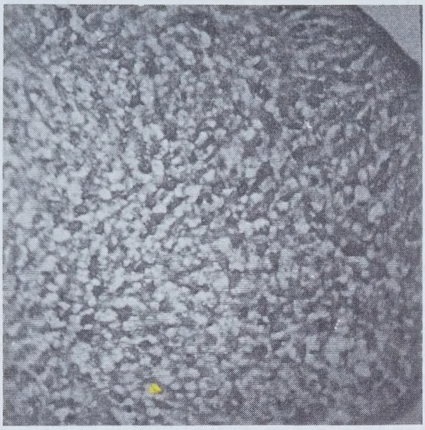


Figure 7: A 256x256 pixel image of the control tissue sample.

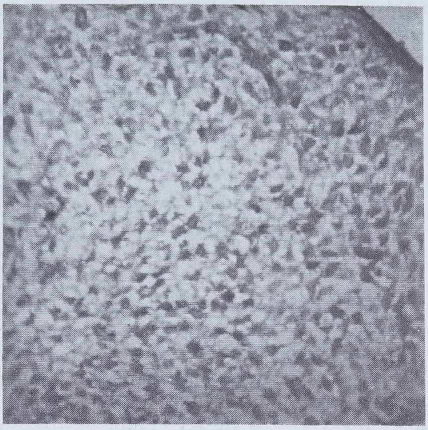


Figure 8: A 256x256 pixel image of the digested tissue sample. This sample slice is consecutive with the sample in Figure 7. This sample has been treated to removed the molecule in question.

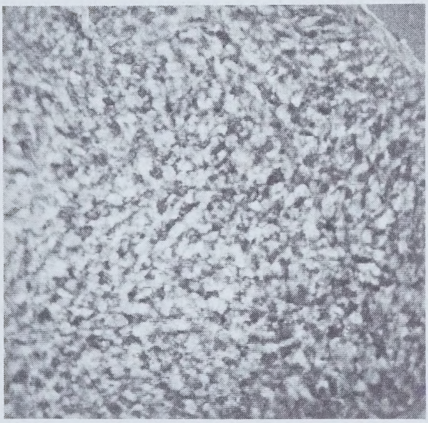


Figure 9: The raw difference image found by subtracting Figure 8 from Figure 7. Notice how the registration noise makes it difficult to see where the images are really different.

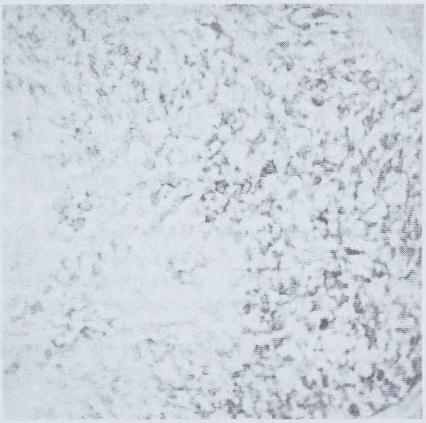


Figure 10: The filtered difference image found by using adaptive gray scale mapping on Figure 9. A neighborhood radius of 7 was used. Notice how it is easier to see the areas where the control image is darker than the digested image.

QUANTITATIVE ANALYSIS OF A MOMENT BASED EDGE OPERATOR[1]

PAUL H. EICHEL and EDWARD J. DELP
Department of Electrical and Computer Engineering
and
Computer, Information, and Control Engineering Program
University of Michigan
Ann Arbor, Michigan

ABSTRACT

Many mathematical operators capable of locating intensity edges in discrete images have been proposed in the literature. We introduce a new operator, based on the sample variance of a group of pixels, which exhibits two unique properties: freedom from a predefined shape and a rigorous stochastic formulation. We demonstrate the utility of the latter property in applying detection theoretic principles to the task of edge detection and in theoretical predictions of experimental performance measures.

INTRODUCTION: EDGE DETECTION IN DIGITAL IMAGES

Detection of intensity edges, that is of local intensity discontinuities at the boundaries of objects, is a low-level process of fundamental importance in image processing. An edge operator is a mathematical operator of small spatial extent which responds in some way to these discontinuities, usually classifying every image pixel as either belonging to an edge or not.

The literature abounds with edge operators [1],[2]. Generally speaking these operators tend to fall in one of three broad classifications: 1) gradient-based; 2) template matching; and 3) parametric models. All of these operators usually employ some global or adaptive threshold on their outputs as a decision device to finally decide whether a given location is on an edge or not.

In this paper, we investigate the use of the sample variance of a local group of pixels as an edge operator. The use of the sample variance in image boundary detection is not new. It has been used for example to detect regions of similar texture, changes in which are defined to be edges[3]. It has also been used as an intensity edge operator [4]. We shall show that given an appropriate image model the Sample - Variance operator has a rather simple probability density function. The problem of edge detection can then be formulated using classical detection theory based on a composite likelihood ratio test. Along with another result which is presented in the Appendix, the optimum threshold for the Pratt Figure of Merit, a common performance measure for edge detectors, can be obtained.

1. DEFINITION OF THE SAMPLE - VARIANCE OPERATOR

The Sample - Variance (SV) operator will be defined as the sample variance of a group of N pixels, i.e.:

$$S^2 = \left\{ \frac{1}{N} \sum_{i=1}^{N} \left[X_i - \frac{1}{N} \sum_{j=i}^{N} X_j \right]^2 \right\} \qquad (1)$$

where: N = number of pixels in operator.
X_i = gray level value of the i^{th} pixel in pixel group.

---

[1] This work was partially supported by the Center for Robotics and Integrated Manufacturing, Robot Systems Division, College of Engineering, University of Michigan and the Air Force Office of Scientific Research under Grant F49620-82-C-0089.

We note that an important feature of this definition is that no predefined operator shape is implied for the edge operator. In fact, the operator is a function of an arbitrary group of pixels. These pixels may be chosen in any manner in the spatial dimensions of a single image, through time in a sequence of images, or across several spectral channels (e.g. Landsat imagery) of image data.

2. IMAGE EDGE MODEL

The image model considered in this paper is a two-dimensional discrete parameter random field [5]. Given a probability space (Ω, A, P), a measurable function, $f_{\vec{r}}(\omega)$; $\omega \in \Omega$; $\vec{r} \in I^2$, is defined on this space as:

$$f_{\vec{r}}(\omega) = g_{\vec{r}} + n_{\vec{r}}(\omega) \quad : \Omega \times I^2 \rightarrow R \tag{2}$$

Here, $g_{\vec{r}}$ is a deterministic discrete image consisting of two or more regions, each of which is of constant gray-level over its extent. The boundaries or edges between these regions have a ramp-like cross section as in Figure 1. The height, width, and direction of a ramp are parameters of a given edge.

To this function $g_{\vec{r}}$ is added a zero mean i.i.d. Gaussian distributed random field $n_{\vec{r}}(\omega)$. For fixed $\vec{r}$, the random variable $f_{\vec{r}}(\omega)$ therefore has the density:

$$p_f(\vec{r}, z) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(z - g_{\vec{r}})^2}{2\sigma^2}} \tag{3}$$

where: σ^2 = noise variance

3. PROBABILITY DENSITY FUNCTION OF THE S.V. OPERATOR

Given the image model of Section 2, the problem then is to apply the SV operator and identify which pixels belong to an edge. We take a detection-theoretic approach to this problem by first finding the probability density function of the SV operator (i.e. find the pdf of S^2 in Equation 1). We can then use one of several well-known tests based on the likelihood ratio to obtain a decision threshold.

Under the assumption of Section 2, each X_i of the SV operator (Equation 1) is a random variable with density function as in Equation 3 i.e. the X_i's are i.i.d. Gaussian random variables with variance σ and means:

$$E\{ X_i \} = g_{\vec{r}_i} \tag{4}$$

where $\vec{r}_i$ is the position vector of the i^{th} pixel. Letting:

$$\bar{X} = \frac{1}{N} \sum_{i=1}^{N} X_i \qquad Y_i = \frac{X_i - \bar{X}}{\sigma} \tag{5}$$

we can rewrite Equation 1 as:

$$\frac{NS^2}{\sigma^2} = \sum_{i=1}^{N} Y_i^2 \tag{6}$$

where:

$$E\{Y_i\} = \frac{1}{\sigma}(E\{X_i\} - \frac{1}{N} \sum_{j=1}^{N} E\{X_j\}) \qquad Var(Y_i) = 1$$

Thus, the variable $\frac{NS^2}{\sigma^2}$ of Equation 8 is the sum of squares of normal, unit variance random variables with unequal means. This is known to have a non-central chi-square distribution of N-1 degrees of freedom [6]. One degree of freedom is lost due to the fact that the sample mean is subtracted from each X_i in Equation 7 instead of the actual mean. The non-centrality parameter, Θ, is given by:

$$\Theta = \sum_{j=1}^{N} (E\{Y_j\})^2 \tag{7}$$

We will denote the density of $\frac{NS^2}{\sigma^2}$ by $X^2_{(\nu,\Theta)}$ i.e. chi-square with ν degrees of freedom and non-centrality parameter Θ.

4. APPROXIMATIONS TO THE DENSITY FUNCTION

The variable $\frac{NS^2}{\sigma^2}$ is distributed as a non-central chi-square. Unfortunately, there is no closed form for the probability density of $X^2_{(\nu,\Theta)}$ [7]. We therefore investigate suitable approximations for the density of $X^2_{(\nu,\Theta)}$.

(1) First, we approximate $X^2_{(\nu,\Theta)}$ by a central chi-square variable, $\rho X'^2_{(\lambda,0)}$, where the constants ρ and λ are chosen so that $X^2_{(\nu,\Theta)}$ and $\rho X'^2_{(\lambda,0)}$ have the same mean and variance [8]:

$$\rho = 1+\frac{\Theta}{\nu+\Theta} \qquad \lambda = \nu+\frac{\Theta^2}{\nu+2\Theta}$$

So:

$$\frac{X^2_{(\nu,\Theta)}}{1+\frac{\Theta}{\nu+\Theta}} \quad \text{is approximately} \quad X^2_{\left(\nu+\frac{\Theta^2}{\nu+2\Theta},0\right)} \tag{8}$$

(2) Second, we approximate the newly formed X'^2 variable as a normal variable. The standard approximation is [8]:

$$[2X'^2]^{1/2} \approx N((2\lambda-1)^{1/2},1) \tag{9}$$

Putting Equations (10) and (11) together, we have:

$$\left[\frac{2NS^2}{\rho\sigma^2}\right]^{1/2} \approx N((2\lambda-1)^{1/2},1)$$

or:

$$S \approx N\left(\left[(2\lambda-1)\frac{\rho\sigma^2}{2N}\right]^{1/2},\frac{\rho\sigma^2}{2N}\right) \tag{10}$$

where:

$$\rho = 1 + \frac{\Theta}{\nu+\Theta} \qquad \Theta = \sum_{j=1}^{N}(E\{Y_j\})^2$$

$$\lambda = \nu + \frac{\Theta^2}{\nu+2\Theta} \qquad \nu = N-1$$

This approximate density for S is used for all calculations of error probabilities, etc. Numerical justification of this approximation has been tabulated by Patnaik[8], Wilson[9] and others. The approximation is fairly good at $N=10$ and converges rather rapidly as N increases.

5. EDGE DETECTION

We now address the original problem of identifying edges in the image. We would like to use our knowledge of the operator's density function to obtain a classification threshold based on a statistical test.

From Equation 10 we see that the SV operator has an approximately normal density function whose mean and standard deviation are a function of the noise power, σ^2, the non-centrality parameter, Θ, and the number of pixels in the operator, N. Θ in turn depends on the proximity of an edge in the operator window. This is illustrated in Figure 2 for a 3x3 window and a ramp edge model with height to width ratio of 2. The contours in Figure 2 are loci of edges with equal parameter Θ, normalized to that of a vertical edge through the center of the window. That is, an edge whose midline is tangent to the 0.5 contour for example has a Θ 0.5 times that of a vertical edge through the center pixel. Thus edges near the center of the window have a high Θ value and the parameter falls off with increasing distance from the center. The response is very similar to the square-root Sobel (3x3) whose magnitude response contours are shown in Figure 3 for comparison.

The mean of the density function of the random variable S however is a function of the parameter Θ. Referring to Equation 10, we see that a large value of Θ (edge present) gives rise to a density function with a large mean as compared to smaller values of Θ (no edge). Thus the density function of S conditioned on the presence or absence of an edge in the operator window has a different mean as in Figure 4. We must chose a decision threshold S_t so as to minimize errors.

The choice of S_t cannot be based on a simple likelihood ratio test because Θ can take on a range of values. Let the H_0 hypothesis be the event, e_0, that no edge is present and H_1 be the event, e_1, that an edge is present. Under H_1 then, Θ may lie in the range $\vartheta \in$ [a,b] where a is the maximum value of Θ (vertical centered edge) and b is appropriately chosen from Figure 2 (e.g. b = 0.5 * a). Under H_0, $\vartheta \in$ (b,0]. i.e. the H_1 hypothesis is that an edge falls inside the b/a contour of Figure 2, the H_0 hypothesis is that the edge falls outside that contour. A composite likelihood ratio test [10] on the variable S for deciding whether or not an edge is present is then:

$$\Lambda(s) = \frac{\int_{\vartheta_0 \in \Omega_0} p(s \mid e_0, \vartheta_0)\, p(\vartheta_0)\, d\vartheta_0}{\int_{\vartheta_1 \in \Omega_1} p(s \mid e_1, \vartheta_1)\, p(\vartheta_1)\, d\vartheta_1} \quad \overset{d_0}{\underset{d_1}{\gtrless}} \quad T \qquad (11)$$

where:

$$\Omega_1 = [a, b] \qquad \Omega_0 = (b, 0]$$

$$d_1 = decide\ e_1\ is\ true \qquad d_0 = decide\ e_0\ is\ true$$

In the case p(ϑ) is not known a priori, one simplifying assumption is that edges are uniformly distributed i.e. no particular edge location or orientation is any more likely than any other. The density is then proportional to the differential area (Figure 2). In any event, Equation 11 can be evaluated numerically to obtain a decision threshold, S_t, for s. If $T = 1$, the resulting test is maximum likelihood. Other possibilities include MAP ($T = \frac{P(no-edge)}{P(edge)}$), Neyman-Pearson, Bayes Risk, and Minimax tests. The probability of error can also be calculated. An example of this procedure will be given in the next section.

6. PREDICTING THE PRATT FIGURE OF MERIT

The Pratt Figure of Merit [11] for edge detectors has been used in the literature as a means for making performance comparisons among edge detectors. It has become a standard experimental tool to quantify the performance of edge detectors so that such comparisons can be made less subjectively. It penalizes for both Type I and Type II errors as well as for displacements of detected edge points from true location. Basically, an artificial image is constructed with a single vertical ramp edge of height h, width two pixels, and with added iid noise of variance σ^2. The edge detector's output from this image is then used to form the figure of merit defined by:

$$F = \frac{1}{I_M} \sum_{i=1}^{I_D} \frac{1}{1 + \alpha l_i^2} \tag{12}$$

where:

$I_M = \max\,(I_D, I_I)$
I_I = number of ideal edge points
I_D = number of detected edge points
l_i = displacement of the i^{th} detected edge point from the ideal edge.
α = scaling parameter

In the past, the merit value for an edge detector was found by experimentally changing the magnitude threshold for the detector until the best merit value was obtained. This experimental optimization had to be performed with a new artificial image for each value of signal-to-noise ratio of interest. But since the SV Edge Operator has a known probability density, we can establish apriori what the Pratt Figure of Merit is as a function of signal-to-noise ratio, and what the optimum threshold on S should be in order to attain that merit value. The ability to do this is crucially dependent on the following Theorem which is proved in the Appendix.

Theorem: When $p(s \mid e_1)$ and $p(s \mid e_0)$ are normally distributed, then the threshold for S that maximizes F is the threshold S_t such that:

$$P(d_0, e_1) = P(d_1, e_0) \tag{13}$$

The corresponding optimum figure of merit is :

$$F \cong \frac{P(d_1, e_1)}{P(d_1)} \tag{14}$$

We will use this Theorem along with the material developed in the previous section to predict the Pratt Figure of Merit for a 3x3 SV operator. We first expand the joint probabilities of Equation 13:

$$P(d_0, e_1) = P(e_1) \int_{S_t}^{\infty} \int_{\vartheta_1 \in \Omega_1} p(s \mid e_1, \vartheta_1)\, p(\vartheta_1)\, d\vartheta_1\, ds$$

$$P(d_1, e_0) = P(e_0) \int_{-\infty}^{S_t} \int_{\vartheta_0 \in \Omega_0} p(s \mid e_0, \vartheta_0)\, p(\vartheta_0)\, d\vartheta_0\, ds \tag{15}$$

Now since the edge is vertical of width two pixels and the operator is a 3x3 square, only three values of Θ are possible. The first corresponds to the window centered on the edge:

$$\vartheta_1 \in \Omega_1 = \{ (3h/2\sigma)^2 \}$$

and the other two to the window one pixel to either side of the edge midline and wholely outside the edge:

$$\vartheta_0 \in \Omega_0 = \{ (h/2\sigma)^2, 0 \}$$

where $\frac{h^2}{\sigma^2}$ is defined to be the signal to noise ratio. Furthermore, if the test image is square of size MxM, these three values of Θ have a priori probabilities in the ratios 1:2:M-2. Consequently, Equations 15 become:

$$
\begin{aligned}
P(d_0,e_1) &= \frac{1}{M}\int_{S_t}^{\infty} p(s|e_1,\vartheta_1=(3h/2\sigma)^2)\,ds \\
&= \frac{1}{M}\,Q\left[\frac{S_t-\mu_1}{\sigma_1}\right] \\
P(d_1,e_0) &= \frac{2}{M}\int_{-\infty}^{S_t} p(s|e_0,\vartheta_0=(h/2\sigma)^2)\,ds \;+\; \frac{M-2}{M}\int_{-\infty}^{S_t} p(s|e_0,\vartheta_0=0)\,ds \\
&= \frac{2}{M}\,Q\left[\frac{\mu_{2a}-S_t}{\sigma_{2a}}\right] \;+\; \frac{M-2}{M}\,Q\left[\frac{\mu_{2b}-S_t}{\sigma_{2b}}\right]
\end{aligned}
\qquad (16)
$$

where μ_1,σ_1 are calculated from Equation 10 for $\vartheta_1=(3h/2\sigma)^2$, etc. Substituting Equation 16 into Equation 13, we have:

$$
Q\left[\frac{S_t-\mu_1}{\sigma_1}\right] = 2\,Q\left[\frac{\mu_{2a}-S_t}{\sigma_{2a}}\right] + (M-2)\,Q\left[\frac{\mu_{2b}-S_t}{\sigma_{2b}}\right] \qquad (17)
$$

which may be solved numerically for the threshold, S_t. Equation 14 is then used to find the resulting Figure of Merit.

Figures 5 and 6 illustrate these results. They compare the theoretical figure of merit and optimum threshold verses snr (solid lines) to the experimental data (stars). The experimental points were obtained by iteratively changing the threshold until the highest figure of merit was obtained, a procedure universal to the literature. The accuracy of the probability model for the SV operator and the above Theorem clearly allows us to predict the outcome with assurance.

7. CONCLUSION

In this paper we have shown that the Sample - Variance operator is useful for detecting intensity edges in discrete images. Under the hypothesis of additive, Gaussian distributed i.i.d. noise, the operator's probability density function may be very accurately approximated by a normal density. This allows the use of a composite likelihood ratio test to determine a decision threshold for the operator. In this way we can implement a variety of classical detection schemes such as Maximum Likelihood and MAP.

We have also proved a theorem concerning predicting the optimum decision threshold for the Pratt Figure of Merit. The predicted optimum threshold for a 3x3 SV operator was found to be virtually identical to the actual value obtained by the usual experimental method. This demonstrates the power of the detection theory approach to finding edges in images.

8. REFERENCES

[1] Kunt, M., "Edge detection: a tutorial review," *Proc. of ICASSP 82. IEEE Inter. Conf. on Acou. Speech & Sig. Proc.* vol.2, 1982.

[2] Davis, L., "A survey of edge detection techniques," *Computer Graphics and Image Processing.*" vol.4, 1975, pp248-270.

[3] Weszka, J., C. Dyer, and A. Rosenfeld, " A comparative study of texture measures for terrain classification," *IEEE Trans. on Sys. Man & Cyber.* vol.6, no.4, April 1976, pp269-285.

[4] Hale, J., "Detection of elementary features in a picture by non-linear local numerical processing," *Proc. Third Int. Joint Conf. on Pattern Recognition.* 1976, pp764-768.

[5] Wong, E., *Stochastic Processes in Information and Dynamic Systems* McGraw Hill, 1971.

[6] Hogg, R. and A. Craig, *Introduction to Mathematical Statistics* Macmillan Co., 1971, pp.316-317.

[7] Tiku, M., "Laguerre series forms of noncentral chi-square and F distributions," *Biometrika* vol.52, 1965, p415.

[8] Patnaik, P., "The noncentral chi-square and F distributions and their applications," *Biometrika* vol.36, 1949, p202.

[9] Wilson, E. and M. Hilferty, "The distribution of chi-square," *Proc. Nat. Acad. Sci.* vol.17, no.694, 1931.

[10] Melsa, J. and D. Cohen, *Decision and Estimation Theory* McGraw-Hill, 1978, pp.21-53.

[11] Pratt, W. and I. Abdou, "Quantitative design and evaluation of enhancement/thresholding edge detectors," *Proc. of the IEEE* vo.67, no.5, May 1979, pp.753-763.

APPENDIX

Theorem: When $p(s \mid e_0)$ and $p(s \mid e_1)$ are normally distributed, then the threshold for S that maximizes F is the threshold S_t such that:

$$P(d_0,e_1) = P(d_1,e_0)$$

This Theorem will be proved in several parts, and only for $F > 0.5$. For F less than this, only a weaker statement can be made.

(i) Lemma (1):
If S_t is chosen such that $P(d_0,e_1) = P(d_1,e_0)$,
Then: $I_M = I_I = I_D$. (See Equation 12)
proof:
Let I_T = total number of image points. Then:

$$I_I = I_T P(e_1) = I_T \left\{ P(d_0,e_1) + P(d_1,e_1) \right\} \tag{18}$$

$$= I_T \left\{ P(d_1,e_0) + P(d_1,e_1) \right\}$$

$$= I_T P(d_1) \qquad = I_D$$

$$\therefore I_M = \max(I_I, I_D) = I_I = I_D$$

Let $F_T \triangleq F$ resulting from $S_t = T$ chosen as above.
Let $F_t \triangleq F$ resulting from threshold $S_t = t$

(ii) Lemma (2): $F_t \le F_T \qquad ; t \ge T$
proof:
$I_D = I_T P(d_1)$ and $P(d_1)$ is a monotonically non-increasing function of t.
Furthermore: $t = T \quad \Rightarrow \quad I_D = I_I$ (Lemma (1))

$$\therefore I_D \le I_I \qquad ; t \ge T$$

$$\therefore I_M = I_I \qquad ; t \ge T$$

Thus:

$$F_t = \frac{1}{I_I} \sum_{i=1}^{I_D} \frac{1}{1 + \alpha l_i^2} \qquad ; t \geq T$$

is a monotonically non-increasing function of t, since I_I is a constant. The lemma follows.

(iii) Lemma (3): $F_t \leq F_T \qquad ; t \leq T \qquad \text{if} \qquad F_T \geq ½$

First, we show that:

$$F_t \cong \frac{P(d_1, e_1)}{P(d_1)} \qquad ; t \leq T \tag{19}$$

proof: $I_M = I_D \qquad ; t \leq T$ (see above). Therefore:

$$F_t = \frac{1}{I_D} \sum_{i=1}^{I_D} \frac{1}{1 + \alpha l_i^2} \qquad ; t \leq T$$

Now, if a detected edge point is a real edge point, $l_i = 0$. $\therefore \frac{1}{1 + \alpha l_i^2} = 1$.

The detected edge points that are not real edge points are randomly scattered throughout the image (iid noise). So $l_i \sim$ large on the average. Therefore:

$$\sum_{false\ edge\ points} \frac{1}{1 + \alpha l_i^2} << \sum_{real\ edge\ points} \frac{1}{1 + \alpha l_i^2}$$
$$= I_T P(d_1, e_1)$$

Therefore:

$$F_t = \frac{1}{I_D} \left\{ \sum_{real\ edge\ points} \frac{1}{1 + \alpha l_i^2} + \sum_{false\ edge\ points} \frac{1}{1 + \alpha l_i^2} \right\}$$
$$\cong \frac{1}{I_T \left\{ P(d_1) \right\}} \left\{ I_T P(d_1, e_1) \right\}$$
$$\cong \frac{P(d_1, e_0)}{P(d_1)} \qquad ; t \leq T$$

Next we prove the lemma by showing that:

$$\frac{dF_t}{dt} > 0 \quad \text{for} \quad t \leq T \quad \text{if} \quad F_T \geq ½ \tag{20}$$

proof: Using Equation 19:

$$\frac{dF_t}{dt} = \frac{d}{dt} \left\{ \frac{P(d_1, e_1)}{P(d_1)} \right\}$$
$$= \frac{1}{\left[P(d_1) \right]^2} \left\{ P'(d_1, e_1) P(d_1) - P'(d_1) P(d_1, e_1) \right\}$$

So:

$$\frac{dF_t}{dt} > 0 \quad \Longleftrightarrow \quad \frac{P'(d_1, e_1)}{P'(d_1)} < \frac{P(d_1, e_1)}{P(d_1)} \tag{21}$$

$$\text{since } P'(d_1) = \frac{d}{dt} P(d_1) < 0$$

$$P'(d_1,e_1) = \frac{d}{dt}\left\{P(d_1,e_1)\right\}$$

$$= \frac{d}{dt}\int_t^{\infty} p(s,e_1)ds \quad = \quad p(s,e_1)$$

And similarly: $P'(d_1) = p(s,e_1) + p(s,e_0)$

$$\therefore \frac{dF_t}{dt} > 0 \iff \frac{p(s,e_1)}{p(s,e_1)+p(s,e_0)} < \frac{P(d_1,e_1)}{P(d_1,e_1)+P(d_1,e_0)}$$

$$\iff \frac{p(s,e_0)}{p(s,e_1)} > \frac{P(d_1,e_0)}{P(d_1,e_1)} \tag{22}$$

Now, from Equation 19:

$$F_t = \frac{P(d_1,e_1)}{P(d_1,e_1)+P(d_1,e_0)}$$

$$\therefore \frac{P(d_1,e_0)}{P(d_1,e_1)} = \frac{1}{F_t} - 1$$

$$\Rightarrow \frac{P(d_1,e_0)}{P(d_1,e_1)} < 1 \quad whenever \quad F_t > ½ \tag{23}$$

Examining Equation 22 in light of Equation 23, we see that Equation 20 and the Lemma will be proved if we can but show:

$$\frac{p(s,e_0)}{p(s,e_1)} \geq 1 \qquad ;t \leq T \tag{24}$$

But by definition, T is such that :

$$P(d_0,e_1) = P(d_1,e_0)$$

$$i.e. \quad \int_{-\infty}^{T} p(s,e_1)ds = \int_{T}^{\infty} p(s,e_0)ds$$

$$or: \quad P(e_1)\int_{-\infty}^{T} p(s\,|\,e_1)ds = P(e_0)\int_{T}^{\infty} p(s\,|\,e_0)ds$$

$$or: \quad P(e_1)Q\left[\frac{\mu_1 - T}{\sigma_1}\right] = P(e_0)\;Q\left[\frac{T-\mu_0}{\sigma_0}\right] \tag{25}$$

Now since $\sigma_1 > \sigma_0$ always for this operator (see Section 5 and Equation 10), Equation 35 therefore implies:

$$\Rightarrow \quad P(e_1)\,p(s=T\,|\,e_1) \quad < \quad P(e_0)p(s=T\,|\,e_0) \tag{26}$$

$$\Rightarrow \quad 1 < \frac{P(e_0)p(s=T\,|\,e_0)}{P(e_1)p(s=T\,|\,e_1)}$$

$$\Rightarrow \quad 1 < \frac{p(s=T,e_0)}{p(s=T,e_1)} \tag{27}$$

$$\Rightarrow \quad 1 < \frac{p(s,e_0)}{p(s,e_1)} \qquad ;\; t \leq T$$

This proves Lemma 3. Lemmas 2 and 3 together prove the Theorem when $F_T > 0.5$. If $F_{\max} \leq 0.5$, only Lemma 2 holds; i.e. the optimum threshold occurs at T or less. Equation 24 gives the resulting Figure of Merit.

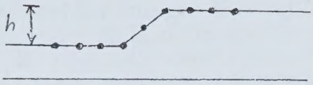


Figure 1: A ramp edge in cross section. Here, the height is h and the width is two pixels.

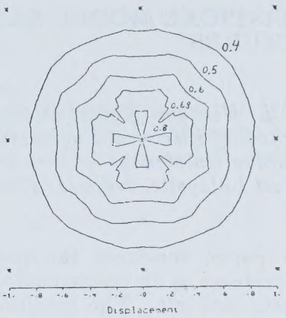


Figure 2: Contours of equal paramet Θ for a 3x3 SV operator. The displac ment is in pixels.

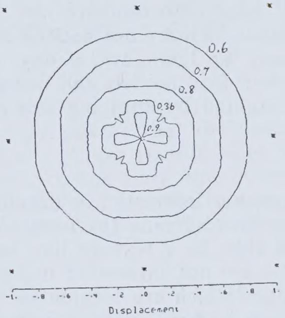


Figure 3: Contours of equal magnitude for the 3x3 Sobel operator.

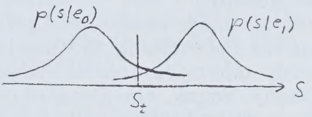


Figure 4: The density function of S conditioned on the presence or absence of an edge in the operator window, with decision threshold S_t.

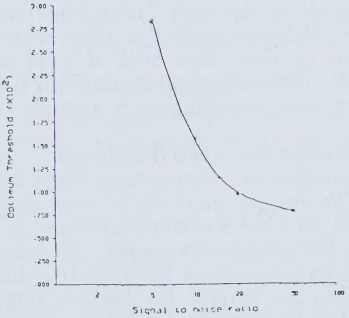


Figure 5: Predicted (curve) verses actual (stars) optimum threshold for the Pratt Figure of Merit.

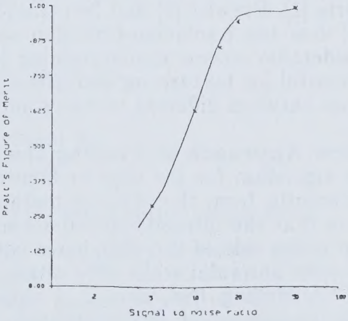


Figure 6: Predicted (curve) verses actual (stars) Pratt Figure of Merit.

A STATISTICAL MODEL BASED APPROACH TO EDGE DETECTION IN TEXTURES[†]

R. L. Kashyap, Robert D. Bauer[*]
School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

Abstract

This paper considers the problem of locating the boundaries between the different textures in an image made up of several textures, usually micro-textures like sand, grass, cork etc. The traditional methods of edge detection are not sufficient since they also yield the micro-edges within a texture, which are not needed here.

The image is divided into a series of strips and the boundary, if any, between the textures in that strip is obtained by fitting random field models and using a likelihood rule. We present the experimental results obtained by applying this rule to a variety of images made up of several textures.

I. Introduction

Given an image made up of several textures, we are interested in developing an algorithm for determining the boundaries between the different textures. We consider mainly micro-textures like sand, grass, raffia etc. In a texture like sand, one can see many 'edges', the so-called microedges. We are not interested in them. An algorithm is desired which will identify only the edges which are boundaries between different textures, ignoring the microedges within each texture, without human intervention.

Currently there are a large number of methods of edge detection. See for example ref [1]. Only two approaches will be mentioned here. In the first approach, the entire image is scanned using gradient and related operators such as those suggested by Roberts [1], Prewitt [2] and Nevatia [3]. The other approach due to Marr and Hildreth [4] uses the Laplacian-Gaussian operator. Whereas both the approaches have had considerable success in determining the edges in some cases, both of them are not very successful for texture segmentation, i.e. both of them cannot distinguish between boundaries between different textures and the microedges within each texture.

II. A New Approach to Finding the Boundaries Between Textures

Our algorithm for the edge or boundary detection is based on modelling an edge quite differently from the existing methods. The current methods of edge detection [1] assume that the intensity distribution around an edge is a step function, i.e., each region on either side of the edge has a constant, but different intensity. This assumption is clearly not valid while with edges caused by a boundary between two different textures. Modelling the edge as a slope is also not helpful for texture boundary detection. A number of studies indicate [5,6] that the correlation pattern in a single texture can be successfully represented by one of the various types of random field models. The boundary between two textures is, thus, a boundary between two regions obeying different random fields.

The image is divided into a large number of horizontal strips of dimension, say, 4×24. The strip size is kept small enough that a edge, if any, can be modelled as a vertical line. Using likelihood arguments, we develop a decision rule for testing whether the window has any edge at all. If the answer is yes, then the column number corresponding to the edge is also estimated. We also test whether the window has more than one edge.

We can repeat the procedure using vertical strips instead of horizontal strips. By a simple algorithm we can combine the boundary information from the two methods and thus obtain the comprehensive boundaries between the various texture.

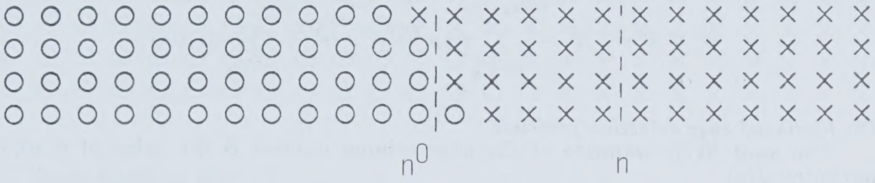
[†] Partially supported by the Office of Naval Research under the grant N00014-82K-0360.

[*] Currently at TRW Inc, One Space Park, Redondo Beach, CA 90277.

We also repeat the results of edge detection by using this rule in a large number of images. Additional details can be found in [7].

III. Model for Images Having Two Textures

Consider an R×C image described by an array of pixel intensities $\{y(s),\ s=(i,j),\ i=1,\dots,R,\ j=1,\dots,c\}$ where $y(s)$ is the intensity of pixel s. Let the image be made up of 2 textures, as shown below with ○ indicating one texture and □, indicating the other.



Let us assume that the boundary between them is vertical and is located at the column n^0. (Later we will relax the assumption about the edge being vertical). n^0 is unknown. Using only the information in he image we will estimate n^0, the edge column number. Let the texture on the left side of the column n^0 obey the following AR model in (3.1) and the texture on the right side obey the model in (3.2).

$$y(s) = (\theta^{(1)})^T z(s) + (\rho^1)^{1/2}\omega(s), \tag{3.1}$$

$$y(s) = (\theta^{(2)})^T z(s) + (\rho^2)^{1/2}\omega(s), \tag{3.2}$$

where $\theta^{(k)} = \text{col.}[\theta^k_\eta, \theta^k_s,\ s\epsilon\Omega_N]$, $k=1,2$, $z(s) = \text{col.}[1, y(s-r),\ r\epsilon\Omega_N]$,

$$(\theta^{(1)})^T z(s) = \sum_r \theta^k_r y(s-r) + \theta^k_\eta .$$

Ω_N is an appropriate set of neighbors around (0,0). In our case, $\Omega_N = \{(-1,1), (-1,-1), (-1,0)\ (0,-1)\}$.

$\omega(\cdot)$ is I.I.D. Normal (0,1) sequence.

$\theta^{(k)}, \rho^{(k)}$, $k=1,2$, are unknown, $\phi^{(k)} = \{\theta^k, \rho^k\}$, $k=1,2$.

Let n be our guess (or estimate) of n^0.

$\Omega_{1,n}$ = The set of pixels to the left of the column n

$\Omega_{2,n}$ = The set of pixels to the right of column n.

$\mathbf{y}_n^{(k)} = \{y(s):\ s\epsilon\Omega_{k,n}\}$, $k = 1,2$. $\mathbf{y} = \mathbf{y}_n^{(1)} \cup \mathbf{y}_n^{(2)}$ = the entire image array.

Let us assume that $\phi^{(1)}$ and $\phi^{(2)}$ have prior probability densities $p_i(\phi^i)$, $i=1,2$, and they are independent. Let us obtain an expression for the joint probability density of the entire image given that n is the *true edge,* namely $p(\mathbf{y}|n)$ or $p(\mathbf{y}_n^{(1)}, \mathbf{y}_n^{(2)}|\ n)$. Note that $p(\cdot)$ involves only the image $\mathbf{y}$ and nothing else. The required 'best' estimate, $\hat{n}$ of n^0 is obtained by maximizing $p(\mathbf{y}|n)$ with respect to n.

We will obtain an explicit expression for $p(\mathbf{y}|n)$ as follows. The probability density of y given to the parameters $\phi^{(1)}$ and $\phi^{(2)}$ can be written easily [7,8].

$$p(\mathbf{y}_n^1, \mathbf{y}_n^2|\ \phi^{(1)}, \phi^{(2)}, n) = p(\mathbf{y}_n^1|\ \phi^{(1)}) p(\mathbf{y}_n^{(2)}|\ \phi^{(2)}) \tag{3.3}$$

$$p(\mathbf{y}_n^{(k)}|\ \phi^{(k)}) \simeq \left(\frac{1}{2\pi\rho^k}\right)^{N_k/2} \exp[-(1/2\rho^k) \sum_{s\epsilon\Omega_{k,n}} (y(s)-(\theta^k)^T z(s))^2]. \tag{3.4}$$

N_k is maximum number of pixels in Ω_k for which z(s) can be computed using y(s), $s\epsilon\Omega_{k,n}$, and the sign '$\simeq$' means 'approximately equal to'. After multiplying (3.3) by $p_1(\phi^{(1)}) p_2(\phi^{(2)})$ and integrating over $\phi^{(1)}$ and $\phi^{(2)}$, we get the desired expression for $p(\mathbf{y}_n^1, \mathbf{y}_n^2|\ n)$. The integration can be carried out explicitly using the concept of asymptotic integration [9,7]. The result is given below.

$\ln p(\mathbf{y}_n^{(1)}, \mathbf{y}_n^{(2)}|\ n) = J(n)$ + terms independent of n, where $J(n) = I + II + III + IV$,

$$I = \sum_{k=1,2} \left\{ -(N_k - m - 2)\ln(\hat{\rho}^k) \right\}, \quad II = \sum_{k=1,2} \left\{ -\ln \det. \left[\frac{1}{N_k} (S^k)^{-1} \right] \right\}$$

$$III = \sum_{k=1,2} \left\{ -(m+1)\ln(N_k) \right\}, \quad IV = \sum_{k=1,2} 2\ln(p(\hat{\phi}^k))$$

$$\hat{\rho}^k = (1/N_k) \sum_{s \epsilon \Omega_{k,n}} (y(s) - (\hat{\theta}^k)^T z(s))^2, \quad \hat{\phi}^k = (\hat{\theta}^k, \hat{\rho}^k)$$

$$\hat{\theta}^k = S^k \alpha^k, \quad S^k = \left(\sum_{s \epsilon \Omega_{k,n}} z(s) z^T(s) \right)^{-1}, \quad \alpha^k = \sum_{s \epsilon \Omega_{k,n}} z(s) y(s)$$

The likelihood edge detection principle:

The most likely estimate of the edge column number is the value of n which maximizes J(n).

Comment 1: If a window has one edge, whether vertical or slanted the graph of J(n) versus n has one dominant peak, as indicated in Figure 2(a),(b). If the window has two edges, then the graph of J(n) versus n indicates two distinct peaks as in figure 2(c). Thus we can detect both the edges in the image, even though the theoretical development used a very restrictive assumption of only one edge in an image.

Comment 2: Consider an image which has *no* edge. The graph of J(n) versus n in that case is given in figure 2(d). Note that the graph is relatively flat. For judging the flatness of the J(n) graph, we use the following criterion. Let the image window be of size 4x24, as in our experiments. Let $h = |\max J(\cdot) - (1/2)(J(5) + J(19))|$.

If the value of n which maximizes J is within 'a' pixels of either end of the window, or if the ratio of the measure h to the standard deviation of J is less than b then there is no edge in the window. The values of a and b are chosen to give good results. The values used here were a = 6 and b = 2.

IV. Experimental Results and Discussion

Various types of images were constructed from natural textures with the number of edges ranging from zero to two. The different textures used in constructing the test images are given in figure 1. All of them were taken from the Brodatz album [10]. Five different experiments were performed using the test images described below.

1) 1000 4×24 images having exactly one vertical edge
2) 1000 4×24 images having one slanted edge
3) 1000 4×24 images windows with exactly 2 vertical edges
4) 400 4×24 images with no edge
5) An 128×128 image made of *several* textures so that the edges cover all directions.

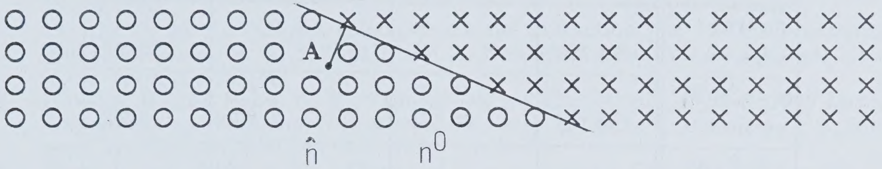
In all the cases, the neighborhood set of the random field model is as indicated in section III.

4.1 Images with one edge (vertical)

1000 windows were constructed with the column number of the true edge lying between 4 and 20. In Figure 2(a), a typical image and the corresponding J(n) versus n graph is demonstrated. Note the strong peak in the J(n) graph near the edge. The distribution of $(\hat{n} - n^o)$ is given in rows 2 and 3 of Table 1.

4.2 Images with one slanted edge

A typical image with one slanted edge is shown in fig. 2(b). As before, the estimated edge is a column number. We need a measure of the deviation of the estimated edge from the true edge. Let n^0 be the column number corresponding to the *middle* of the actual slanted edge. Then the distance between the true (slanted) edge and $\hat{n}$ is defined as the distance from point A to the line representing the true edge as in the figure below. Point A is the point on the line between rows 2 and 3 of the window on the line representing the most likely edge. Since the slope of the line representing the true edges is 1/2 for these trials the distance between the most likely edge and the true edges is $\frac{1}{\sqrt{5}} \left| n^0 - \hat{n} \right|$. The corresponding results with 1000 trials are summarized in rows 4 and 5 of table 1.



4.3 *Images with no edge*

A typical image no edge along with its J(n) graph is given in figure 2(d). We use the criterion given earlier for deciding whether no edge is present in the image. Of the 400 trials, the decision rule indicated in 74.7% of the images that there was no edge.

4.4 *Images with two edges*

The theoretical development of the decision rule assumed that there is at most one edge in the image. However, the rule will work even if there are 2 edges. Figure 2(c) shows the image and the corresponding J(n) graph. The +′s, O s, and x′s each represent a different region. The true edges are at n^1 and n^2. We can see that J_2 has relative maxima at each true edge. For windows with n^1 and n^2 closer than about 7 pixels only one relative maximum is typically seen in the log likelihood function and it is usually very close to either n^1 or n^2. We constructed 1000 trials with two true edges anywhere between columns 4 and 20 in the window. In only 2.1% of the trials was an edge detected more than 4 pixels away from one of the true edges. In 39.2% of the trials no edge was detected.

4.5 *Summary of results for detecting the most likely edge in a window*

Table 1 summarizes the results. The scheme is very good at detecting an edge which is between columns 6 and 18 of the window and is closer to being vertical than horizontal. If the true edge is any closer to one end of the window or if it is too near to being horizontal then the scheme is not likely to detect the edge. For windows with no actual edge an edge will be detected about 1/4 of the time. This is the weakest aspect of the results presented here. If we had increased the size of the window the rate of false edge detection would have gone down at the expense of higher computing cost. The strongest aspect is that if an edge does exist the scheme usually finds it.

4.6 *Finding the texture boundaries in a general image*

Consider a general image as in figure 3(a) made of several textures. We will divide the image into strips, find the texture boundary in each window and integrate the result. The procedure is indicated in the algorithm given below.

Algorithm to find boundaries between textures in an arbitrary image

```
i)   Position window in the upper left corner of the image.
ii)  Find the most likely vertical edge in the window. Record as a likely vertical
     edge the pixel which is in the second row of the window and in the column
     found most likely to be a vertical edge.
iii) If the window is not positioned to the far right of the image
          then move the window to the left by six pixels,
          else if the window is not at the bottom of the image
```

Table 1. Results of the edge detection scheme on trial windows made of textures.

	Detection of Edges in Windows			
	no edge detected	$\lvert n^0-\hat{n}\rvert \le 2$	$3 \le \lvert n^0-\hat{n}\rvert \le 4$	$4 \le n^0-\hat{n}\rvert$
no true edge	74.7%	-	-	25.5%
$4 \le n^0 \le 20$ vertical	34.1%	60.5%	3.3%	2.2%
$6 \le n^0 \le 18$ vertical	23.5%	70.0%	4.3%	2.4%
$4 \le n^0 \le 20$ slope=1/2	44.4%	51.6%	3.0%	1.0%
$6 \le n^0 \le 18$ slope=1/2	36.7%	59.4%	3.0	0.9%
$4 \le n^0 \le 20$ slope=1/4	72.1%	12.1%	15.8%	0.0%

then move the window down one pixel and position it to the far left of the image,
else goto v)

iv) goto ii)
v) Save the results of steps i) through iv) as the set of likely vertical edges.
vi) Transpose the image and repeat steps i) through iv) on the transposed image. Transpose the result and save it as the set of likely horizontal edges.
vii) Pass the set of likely vertical edges and likely horizontal edges through a relaxation routine to extract the edges in the image.

Steps i) through iv) constitute the actual edge detection scheme. The horizontal window used in steps i) through iv) does not allow edges which are more horizontal than vertical to be detected. Step vi) is equivalent to passing a vertical window over the image which allows horizontal edges to be detected. Step ii) records the pixel in the second row of the window and in the column found most likely to be a vertical edge because that pixel seems to be closest to the true edge in the window using the distance measure suggested earlier. The window is moved over the image as described in step iii) so that each pixel is at the second row and between columns 6 and 18 for two positions of the window. This was done because an edge is more likely to be detected if it is between columns 6 and 18 of the window and so that we give each pixel two "chances" to be detected as an edge. We could have allowed more chances for an edge to be detected by not moving the window over as far each time. Doing this would give the relaxation routine more information to work with, improving the overall results at the expense of more computation time.

Figure 3 shows the results of the scheme applied to an image constructed from textures shown in Figure 1. Figure 3(a) shows the original image. It is 128 pixels by 128 pixels with grey levels. Figure 3(b) shows the set of likely vertical edges represented by the white pixels. Although the relaxation scheme is not part of the work presented here, we apply some very simple relaxation to obtain Figure 3(c). We call Figure 3(c) the set of "very" likely vertical edges, (for lack of a better name). If a pixel in Figure 3(b) representing a likely vertical edge is surrounded by no other likely vertical edges, i.e. a white pixel surrounded by all black pixels, then it is not included in the set of very likely vertical edges. Figure 3(d) is the superposition of *very likely vertical edges* and *very likely horizontal edges,* which we call simply the set of very likely edges. It is clear that figure 3(d) captures all the major texture boundaries in the original image in figure 3(a).

References

[1] A. Rosenfeld and A. C. Kak, *Digital Picture Processing,* New York: Academic Press, 1976.

[2] Prewitt, J. M. S., "Object enhancement and extraction," in *Picture processing and psychopictorics,* R. S. Lipkin and A. Rosenfeld, Eds., Academic Press, New York, 1970, pp. 75-149.

[3] Nevatia, R. and Babu, K. R., "Linear feature extraction," in *Proc. 1978 Image Understanding Workshop,* (Pittsburgh, Pa., May 1978), L. S. Baumann, Ed., Science Applications, McLean, Va., 1978, pp. 73-78.

[4] Marr, D. and Hildreth, E., "Theory of edge detection," *Proc. R. Soc. London B,* vol. 297, (1980), 187-217.

[5] R. L. Kashyap, "Analysis and synthesis of image patterns by spatial interaction models," in *Progress in Pattern Recognition,* (Ed.) L. N. Kanal and A. Rosenfeld, Vol. 1, North Holland, 1981, pp. 149-186.

[6] R. L. Kashyap and R. Chellappa, "Estimation and Choice of Neighbors in Spatial Interaction Models," *IEEE Trans. on Information Theory,* Vol. IT-29, Ja. 1983, pp. 60-72.

[7] Robert D. Bauer, "Edge detection in images using the maximum likelihood principle," M.S. Thesis, School of Electrical Engineering, Purdue University, West Lafayette, IN 47907, August 1983.

[8] E. J. Delp, R. L. Kashyap, and O. R. Mitchell, "Image Data Compression Using Autoregressive Time Series Models," *Pattern Recognition,* vol. 11, pp. 313-323, 1979.

[9] R. L. Kashyap, "Optimal Choice of AR and MA Parts in Autoregressive Moving Average Models," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. PAMI-4, pp. 99-104, March 1982.

[10] P. Brodatz, *Textures: A Photographic Album for Artist and Designers,* New York: Dover Publication, 1966.

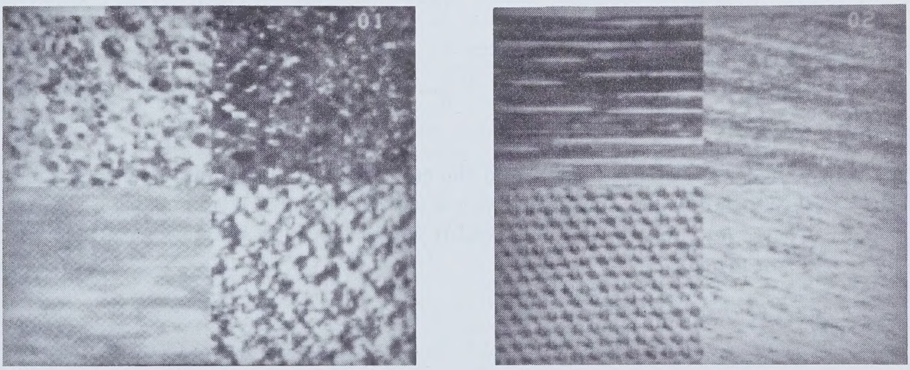


Figure 1: Textures from [10] used in the windows for experiments of section IV.

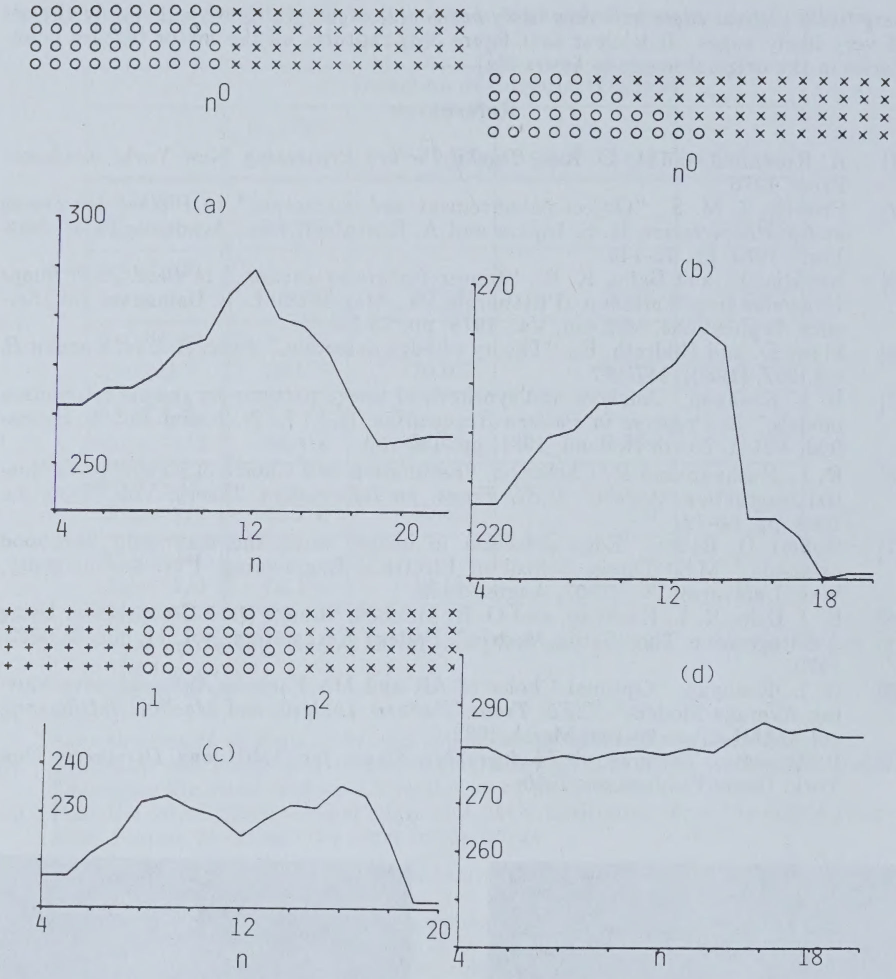
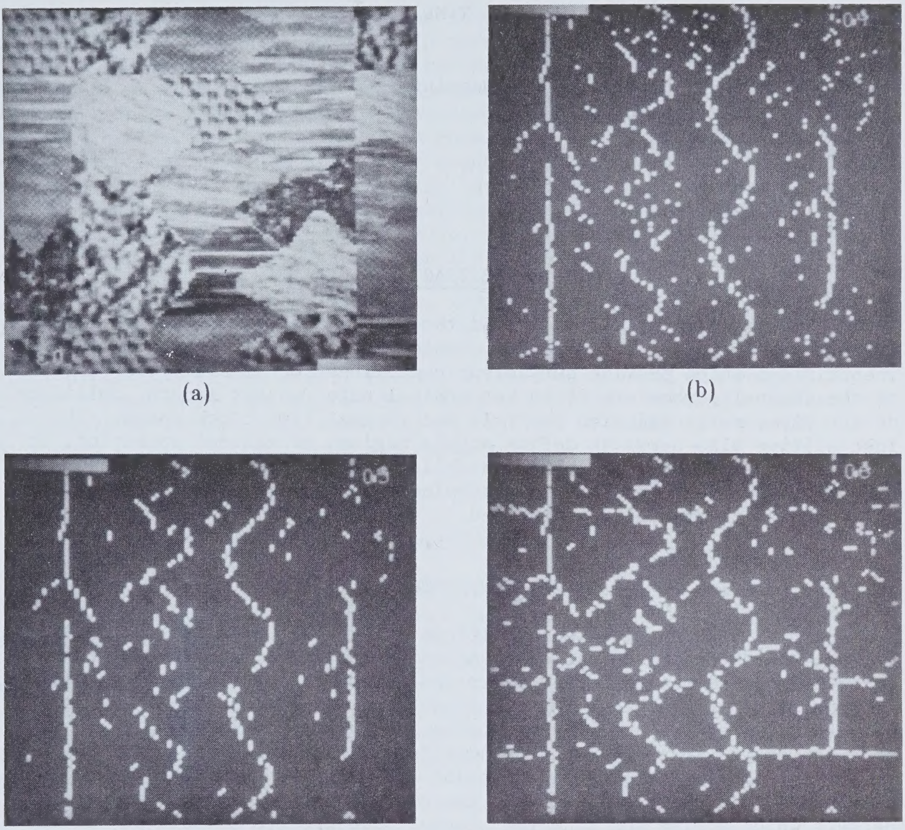


Figure 2. The 4×20 image window and the corresponding graph of J_2 (n) vs. n for 4 different cases. (a) window with a vertical edge.(b) window with a slanted edge (c) window with 2 edges.(d) window with no edge, hence window is not displayed.



(a) (b)

(c) (d)

Figure 3: (a) An image constructed from several textures in [10], (b) The set of likely vertical edges found by our algorithm, (c) The set of "very likely" vertical edges obtained from (b); (d) The superposition of the 'very likely' vertical boundaries and "very likely" horizontal boundaries for the image in (a).

STABILITY AND THROUGHPUT IN VIRTUAL TIME CSMA[†]

J. S. Meditch
Department of Electrical Engineering
University of Washington
Seattle, WA 98195

C.-T. A. Lea
Bell Laboratories
Naperville, IL 60566

ABSTRACT

A new approach for the stability and throughput analysis of Virtual Time CSMA is developed and illustrated. Basic results are in the form of inequalities which provide sufficient conditions for stability as functions of the channel parameters of packet arrival rate, packet length, collision detect time, retransmission control, and virtual time clock speed. The inequalities also serve to define stable regions of channel operation, to demonstrate robustness in channel stability in the presence of channel parameter uncertainties, and to determine the range on achievable average throughput with stability guaranteed.

1. INTRODUCTION

Virtual Time Carrier Sense Multiple Access (VT-CSMA) was introduced in [1] as a computer-communication broadcast channel multiaccess protocol which offers certain advantages over the other members of the CSMA family of protocols. Principal among the advantages was the fact that new data packets which arrive and find the channel busy are not immediately rescheduled for transmission at random for a later time. Rather, they hold their position in the channel queue and are "clocked" out onto the channel in the order of their arrival when the channel becomes idle. New data packets which arrive and find the channel idle proceed immediately in the same way. While this does not preclude the occurrence of collisions which are due to the simultaneous or near simultaneous arrival of two or more packets, it does provide fairness of initial transmission on a first-come/first-serve basis. The above procedures also apply to collided packets which are scheduled for retransmission at a later time. As shown in [1], VT-CSMA has a lower collision rate and better delay-throughput behavior than the other CSMA protocols. However, the study in [1] did not explicitly treat the issue of the stability of VT-CSMA. Rather, it was assumed that all collided packets would be retransmitted successfully via a uniform random retransmission policy for the infinite population channel or by a geometrically distributed one for the finite population case.

In this paper, we provide a detailed study of the stability of VT-CSMA and the throughput behavior which is associated with stable VT-CSMA. We consider the infinite population, slotted case in which the channel parameters are the new packet arrival rate, packet length, collision detect time, retransmission control, and virtual time clock speed. To avoid unnecessary repetition of details, we generally refer to this case in the sequel simply as VT-CSMA.

[†] Research sponsored in part by the National Science Foundation under NSF Grant ECS-8011262.

The contents of the paper are as follows. In Section 2, we develop a two-dimensional imbedded Markov chain model for VT-CSMA from which we derive a one-dimensional model that suffices for our stability and throughput analysis. The latter is presented in Section 3 where we define VT-CSMA to be stable if the above one-dimensional chain is ergodic. Application of Pakes' lemma [2] then results in a pair of inequalities whose satisfaction yields sufficient conditions for stability as functions of the five VT-CSMA channel parameters introduced above. These inequalities provide for a variety of parametric stability tradeoff studies which we illustrate via three examples. A key finding is the existence of stable regions of operation for VT-CSMA over its parameters. This implies a robustness in channel stability in the presence of channel parameter variations and/or uncertainties. The effect is quite dramatic for VT-CSMA with collision detection relative to the case where collision detection is absent. The throughput characteristics for VT-CSMA are then developed, first directly from the stability inequalities and second via renewal theory. From these results, we determine the maximum throughput which is achievable by VT-CSMA with stability guaranteed. Finally, we conclude in Section 4 with a discussion of our results and an indication of some related issues for future research.

2. MATHEMATICAL MODEL

Under the VT-CSMA protocol, each station is equipped with two clocks C and C', and one register V which maintain real time, virtual time, and packet arrival time, respectively. Initially, we assume that C and C' are set to zero and begin running continuously at the same speed. Upon sensing the channel busy, the virtual time clock stops, but the real time clock continues running. When the channel becomes idle, the virtual time clock resumes running at a speed $\eta > 1$ times faster than the real time clock. When it catches up with the real time clock, the virtual time clock then runs at the same speed as the latter. Whenever a station generates a packet, the arrival time is noted in register V.

In slotted VT-CSMA, the time axis is divided into minislots whose duration is the channel's one-way propagation delay. Stations which have a packet ready for transmission proceed as follows. When virtual time C' reaches the packet's arrival time V, the packet is transmitted at the beginning of the next minislot. In viewing packet transmissions, we can think in terms of a window whose size is $w = \min(\eta, C-C')$ minislots moving along the virtual time axis when the channel is silent. Each time the real time clock advances 1 minislot, the virtual time clock will advance w minislots, and the window on the virtual time axis moves forward one step. All packets whose arrival times are within the same window are transmitted at the beginning of the next minislot. A collision occurs if two or more packets are transmitted simultaneously, and collisions are resolved via a suitable retransmission policy.

We now formulate the imbedded Markov chain model for VT-CSMA that provides the basis for our stability and throughput investigation. In our model, we assume that the minislot length is τ sec, that time is expressed in units of minislots, and that packets have fixed length $T > 1$ minislots. Further, we assume that the system is comprised of a large number of stations which collectively generate new packets in a Poisson stream with average rate λ packets/minislot. We consider VT-CSMA both with and without a collision detection capability. The duration of a collision transmission is then T and $R < T$ minislots, respectively. Hence, in general, we have $R \leq T$. Stations which have a packet ready for transmission are divided into two groups: blocked and waiting. Blocked stations are those whose packets were transmitted and collided with others. A blocked station will transmit its packet during the current window with retransmission probability $0 < f \leq 1$. Waiting stations are those whose packet arrival times V are within the range $C' < V < C$. Such packets are, therefore, awaiting their initial transmission. The total traffic which is offered during the current window on the virtual time axis then consists of that from both the blocked and waiting stations.

In considering channel activity, we have already observed that the window $w = \min(\eta, C-C')$ moves forward one step on the virtual time axis as the real time clock advances one minislot. Within this window, three mutually exclusive events can occur: no arrivals, one arrival, and two or more arrivals. If there is only one arrival, then a successful transmission will occur at the beginning of the following minislot and will last T+1 minislots (T minislots to transmit the packet and one minislot to clear the channel due to propagation delay). Therefore, this event occupies T+2 minislots. In the same way, R+2 minislots correspond to the event wherein there are two or more arrivals. If there are no arrivals, the duration of the event is one minislot. Imbedded time is chosen immediately at the end of these events. We let t_k represent the kth imbedded time and illustrate a snapshot of the channel in Fig. 1 for the case where $T = 9$, $R = 3$, and $\eta = 4$.

To specify the state of the channel, we need the two-dimensional state variable (D,B) which represents the current delay in virtual time $D = C-C'$, and the number of blocked stations B, respectively. Both D and B are integers, and the units on D are minislots. We let $(D,B)^k$ represent the channel state at the kth imbedded time, and defind $P = [P_{ij,mn}]$ to be the probability transition matrix from t_k to t_{k+1}. In particular, we have

$$P = [P_{ij,mn}] = \mathrm{Prob}[(D,B)^{k+1} = (m,n) \mid (D,B)^k = (i,j)] \qquad (1)$$

where i, j, m, n ≥ 0.

Determination of the $P_{ij,mn}$, although tedious, is relatively direct. In particular, we recall that channel traffic is generated from both the blocked and waiting stations, and that each blocked station will transmit during the current window with probability f. Also, we note that traffic from the waiting stations is Poisson with average rate λw packets/window step where $w = \min(\eta, D)$. It then follows that

$$P_{ij,mn} = \begin{cases} (1-f)^j (e^{-\lambda w}) & (m,n) = (i+1-w, j) \\ jf(1-f)^{j-1}(e^{-\lambda w}) & (m,n) = (i+T+2-w, j-1) \\ (1-f)^j (\lambda w e^{-\lambda w}) & (m,n) = (i+T+2-w, j) \\ [1-jf(1-f)^{j-1} - (1-f)^j](e^{-\lambda w}) & (m,n) = (i+R+2-w, j) \\ [1-(1-f)^j](\lambda w e^{-\lambda w}) & (m,n) = (i+R+2-w, j+1) \\ \dfrac{(\lambda w)^k (e^{-\lambda w})}{k!} & (m,n) = (i+R+2-w, j+k),\ k \geq 2 \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

where $i = 1, 2, \cdots, j = 0, 1, 2, \cdots$, and $w = 0, 1, 2, \cdots$. The results for $i = 0$ are of a similar form.

It is clear that the stability of VT-CSMA depends upon the behavior of (D,B) as a function of time. Equivalently, we see that it is sufficient to consider the behavior of max(D,B), and, therefore, introduce the state variable transformation

$$X = \max(D,B) \tag{3}$$

This reduces the above two-dimensional Markov chain to a one-dimensional chain for which the state space is $\{0,1,2,\cdots\}$ as indicated in Fig. 2. It can be shown that this latter chain is irreducible and aperiodic. While the corresponding probability transition matrix may be determined using Eq. (2), this is not explicitly necessary for present purposes.

3. STABILITY AND THROUGHPUT

In this section, we study the stability and throughput properties of VT-CSMA following a procedure similar to that in [3]. We begin with the following

Definition. VT-CSMA is stable if the Markov chain specified by Eq. (3) is ergodic.

Sufficient conditions for stability are then provided by

Pakes' Lemma [2]. Let $\{X^k\}$ where $k = 0, 1, \cdots$ indexes time be an irreducible, aperiodic Markov chain whose state space is the set of non-negative integers. The following conditions are sufficient for the chain to be ergodic:

$$\left| E[X^{k+1} - X^k \mid X^k=i] \right| < \infty \tag{4a}$$

and

$$\lim_{i \to \infty} \sup E[X^{k+1} - X^k \mid X^k=i] < 0 \tag{4b}$$

where $E[\cdot|\cdot]$ denotes conditional expectation.

Since $X^k = \max(D,B)^k$, we see that two cases must be considered in attempting to satisfy Eq. (4b), viz., (i) $D \to \infty$, B = any value, and (ii) D = any value, $B \to \infty$. With respect to these two cases, we therefore have $X^k = \max(D,B)^k = D^k$ and $X^k = \max(D,B)^k = B^k$, respectively. The resulting relations to be satisfied are thus

$$\lim_{i \to \infty} \sup E[D^{k+1}-D^k \mid D^k=i,B^k=j] < 0 \tag{5}$$

and

$$\lim_{j \to \infty} \sup E[B^{k+1} - B^k \mid D^k=i,B^k=j] < 0 \tag{6}$$

respectively.

In the sequel, we assume that λ, T, and R are given, and consider retransmission policies in which $f = \alpha/j$ where $j \geq 1$ is the number of blocked stations B and $\alpha > 0$ is the retransmission parameter. We remark that the choice of retransmission policies here is the same as in [3] where the infinite population, nonpersistent CSMA and CSMA/CD protocols were studied. Here, however, stability is a function of the pair (α,η) rather than of α alone.

With the aid of Eq. (2), it can be shown that

$$\begin{aligned} E[D^{k+1} - D^k \mid D^k=i, B^k=j] &= [\sum_s \sum_t s\, P_{ij,st}] - i \\ &= -w + (T+2)[jf(1-f)^{j-1}e^{-\lambda w} + (1-f)^j \lambda w e^{-\lambda w}] \\ &\quad + (R+2)[1-(1-f)^j e^{-\lambda w} - jf(1-f)^{j-1}e^{-\lambda w} - (1-f)^j \lambda w e^{-\lambda w}] \\ &\quad + (1-f)^j\, e^{-\lambda w} \end{aligned} \tag{7}$$

and

$$\begin{aligned} E[B^{k+1} - B^k \mid D^k=i, B^k=j] &= (\sum_s \sum_t n\, P_{ij,st}) - j \\ &= \lambda w - jf(1-f)^{j-1}e^{-\lambda w} - (1-f)^j \lambda w e^{-\lambda w} \end{aligned} \tag{8}$$

respectively, where $w = \min(\eta,D)$. Setting $f = \alpha/j$ in these two expressions, we find that the right-hand sides are finite for all finite λ, T, R, f, w, i, j, and therefore that Eq. (4a) of Pakes' lemma is satisfied. Further, taking the limits indicated in Eqs. (5) and (6), we obtain the following two sufficient conditions for stability,

$$\begin{aligned} \limsup_{i \to \infty} E[D^{k+1}-D^k \mid D^k=i,\ B^k=j] = \{&-\eta+(T+2)[\alpha(1-\alpha/j)^{j-1}e^{-\lambda\eta}+(1-\alpha/j)^j\lambda\eta e^{-\lambda\eta}] \\ &+ (R+2)[1-(1-\alpha/j)^j e^{-\lambda\eta}-\alpha(1-\alpha/j)^{j-1}e^{-\lambda\eta} \\ &- (1-\alpha/j)^j\lambda\eta e^{-\lambda\eta}] + (1-\alpha/j)^j e^{-\lambda\eta}\} < 0 \end{aligned} \tag{9}$$

and

$$\limsup_{j \to \infty} E[B^{k+1} - B^k \mid D^k=i,\ B^k=j] = [\lambda w-\alpha e^{-\alpha}e^{-\lambda w}-e^{-\alpha}(\lambda w)(e^{-\lambda w})] < 0 \tag{10}$$

respectively. Owing to the complexity of the model, Eqs. (9) and (10) are as far as we can go. Therefore, in order to proceed, we make two simplifying assumptions which we believe are reasonable for a number of cases of interest. First, we assume that $D > \eta$ which implies that $w = \min(\eta,D) = \eta$. This corresponds to changing our protocol slightly by restricting the window on the virtual time axis to move only when $D \geq \eta$. Second, we let $e^{-\alpha}$ approximate $(1-\alpha/j)^j$ for $j \geq 1$ and $\alpha << 1$. The accuracy of this approximation is readily checked. Utilizing these two approximations in Eqs. (9) and (10), we obtain

$$(T+2)Ge^{-G} + (R+2)(1-Ge^{-G}-e^{-G}) + e^{-G} < \eta$$

and

$$\lambda\eta < Ge^{-G}$$

respectively, where $G \triangleq \alpha + \lambda\eta$ is the average offered traffic in the current window. Finally, combining the above two relations yields

$$\frac{Ge^{-G}}{\lambda} > \eta > (T+2)Ge^{-G} + (R+2)(1-Ge^{-G}-e^{-G}) + e^{-G} \tag{11}$$

Equation (11) permits us to examine a variety of stability tradeoff cases over the VT-CSMA parameter set $(\lambda, T, R, \alpha, \eta)$. Before illustrating this below, it will prove convenient to introduce the quantity $\lambda^* \triangleq \lambda T$ which is the average arrival and throughput rate in packets/(full) slot when the channel is stable, and satisfies $0 \leq \lambda^* \leq 1$.

Example 1.

(a) Let T = 20, R = 20, and $\lambda^* = 0.6$ which corresponds to $\lambda = 0.03$ packets/minislot. The curves 1 and 2 in Fig. 3 are plots of the left- and right-hand sides in Eq. (11), respectively, as functions of G. Hence, the shaded region in the figure defines the set of all (G,η), equivalently (α,η) since $G = \alpha + \lambda\eta$, for which this VT-CSMA channel is stable.

(b) For T = 20, R = 20, $\alpha = 0.05$, and $\eta = 5$, Eq. (11) becomes

$$\frac{Ge^{-G}}{\lambda} > 5 > 22(1-e^{-G}) + e^{-G}$$

where $G = 5\lambda + 0.05$. This relation is satisfied if $0 < \lambda < 0.03226$, or equivalently when $0 < \lambda^* < 0.6452$.

Part (a) demonstrates a robustness in the stability of the channel over the parameter set (α,η) at constant channel throughput. For instance, if $\eta = 10$, the channel is stable for $0.19 < \alpha < 0.26$, and similarly in η for a fixed α. Alternately, we may select the center of mass of the shaded region in the figure as a "good" operating point in (α,η) for T = 20, R = 20, $\lambda^* = 0.6$. Part (b) of the example demonstrates a robustness in channel stability over the range of average throughputs $0 < \lambda^* < 0.6452$ for fixed T, R, α, and η.

The next example demonstrates the effects of collision detection on channel stability with all parameters the same as above except R.

Example 2.

(a) Let T = 20, R = 5, and $\lambda^* = 0.6$. The stable region of operation over (G,η), equivalently over (α,η), is shown in Fig. 4. The improvement which collision detection affords, particularly with respect to stability margins in the presence of parameter uncertainties, becomes very clear upon comparing Figs. 3 and 4.

(b) For T = 20, R = 5, $\alpha = 0.05$, and $\eta = 5$, Eq. (11) is

$$\frac{Ge^{-G}}{\lambda} > 5 > 22Ge^{-G} + 7(1-Ge^{-G}-e^{-G}) + e^{-G}$$

where $G = 5\lambda + 0.05$. Stable operation then requires $0 < \lambda^* < 0.7280$, a 12.8% increase in the stable throughput range over that in Example 1.

The results of a third and final example are given in Fig. 5 in which the double cross-hatched area defines the stable (G,η) region when $T = 100$, $R = 100$, and $\lambda^* = 0.6$; and the entire shaded area is the stable region for $T = 100$, $R = 10$, and $\lambda^* = 0.6$. The advantages of collision detection are quite dramatic in this case.

We turn now to the problem of maximizing throughput for VT-CSMA subject to Eq. (11), i.e., subject to stability. First, we introduce the following

Proposition. Let $G = \alpha + \lambda\eta$. Then there exist (α,η) for which Eq. (11) holds if and only if

$$\lambda < \frac{Ge^{-G}}{(T+2)(Ge^{-G}) + (R+2)(1-Ge^{-G}-e^{-G}) + e^{-G}} \tag{12}$$

Proof. Suppose that

$$\lambda < \frac{Ge^{-G}}{(T+2)(Ge^{-G}) + (R+2)(1eGe^{-G}-e^{-G}) + Ge^{-G}}$$

Then it is obvious that

$$\frac{Ge^{-G}}{\lambda} > (T+2)(Ge^{-G}) + (R+2)(1-Ge^{-G}-e^{-G})$$

Therefore, we can always find an η such that

$$\frac{Ge^{-G}}{\lambda} > \eta > (T+2)(Ge^{-G}) + (R+2)(1-Ge^{-G}-e^{-G}) + e^{-G}$$

Once η is determined, the corresponding α is given by $\alpha = G - \lambda\eta$.

On the other hand, if Eq. (11) holds, then there exist α and η such that

$$\frac{Ge^{-G}}{\lambda} > \eta > (T+2)\ Ge^{-G} + (R+2)(1-Ge^{-G}-e^{-G}) + e^{-G}$$

from which it follows that

$$\frac{Ge^{-G}}{\lambda} > (T+2)\ Ge^{-G} + (R+2)(1-Ge^{-G}-e^{-G}) + e^{-G}$$

Rearranging this relationship, we have

$$\frac{Ge^{-G}}{(T+2)(Ge^{-G} + (R+2)(1-Ge^{-G}-e^{-G}) + e^{-G}} > \lambda$$

which concludes the proof.

From the Proposition, we can infer the maximum throughput of a stable VT-CSMA channel for fixed T and R. Specifically, multiplying both sides in Eq. (12) by T, we find that

$$\lambda^* < \frac{TGe^{-G}}{(T+2)(Ge^{-G}) + (R+2)(1-Ge^{-G}-e^{-G}) + e^{-G}} \tag{13}$$

where $\lambda^* = \lambda T$ is the average arrival rate in packets/(full) slot. When the channel is stable, λ^* also equals the average throughput. For a given G, the corresponding value of the right-hand side in Eq. (13) defines the upper limit L^* on the packet arrival rate below which, according to the Proposition, we can always find an (α,η) for which the channel is stable. By plotting the right-hand side in Eq. (13) as a function of G for fixed values of T and R, we not only obtain L^* as a function of G, but also the corresponding region (G,λ^*) of stable channel operation. This is illustrated in Fig. 6 for the four cases where T = 20, R = 20; T = 20, R = 2; T = 100, R = 100; and T = 100, R = 2. The advantages of including collision detection are also apparent here, noting in particular the relative insensitiviey of L^* as a function of G for the longer packet length with short collision detection time.

The interpretation of Eq. (13) also follows from renewal theory. Assuming that the total traffic during each window is Poisson with intensity G, then three events can occur after the window advances one step: no arrivals, one arrival, or two or more arrivals. Now let V be the expected duration of these events and U be the average time during an event within which the channel is used without conflict. From renewal theory, the average channel utilization is

$$S = \frac{U}{V} \tag{14}$$

For our model, we have

$$V = (T+2)(Ge^{-G}) + (R+2)(1-Ge^{-G}-e^{-G}) + e^{-G} \tag{15}$$

and

$$U = TGe^{-G} \tag{16}$$

Substituting Eqs. (15) and (16) into Eq. (14) yields

$$S = \frac{TGe^{-G}}{(T+2)(Ge^{-G}) + (R+2)(1-Ge^{-G}-e^{-G}) + e^{-G}} \tag{17}$$

whose right-hand side is identical to that in Eq. (13). This implies that if the arrival rate is less than the average utilization of the channel, then the channel is stable. The value G_{max} which maximizes the right side in Eq. (13) hence maximizes the average channel throughput, and consequently defines the optimum operating point for the channel. For example, if T = 20 and R = 20, we get $G_{max} = 0.241$. Assuming that the arrival rate is $\lambda^* = 0.6$, we find from Eq. (11) that the channel is stable for $5.497 < \eta < 6.313$. Setting $\eta = 6$ gives $\alpha = G - \lambda\eta = 0.061$. Hence, channel utilization is optimized for $(\alpha,\eta) = (0.061,6)$.

We conclude this section by showing the maximum achievable throughput $\lambda^*_{\max}$ for stable VT-CSMA as a function of R for a number of fixed values of T. The results are given in Fig. 7. We remark that the throughput results developed here, namely these in Eqs. (13) and (17) and in Figs. 6 and 7, are identical to those obtained in [3] for the stable, infinite population, slotted, nonpersistent CSMA and CSMA/CD protocols. Earlier results noting that VT-CSMA and nonpersistent CSMA have the same throughput characteristics were given in [1]. While the results in [1] follow via renewal theory, the ones here are a consequence of our stability analysis and the Proposition. Our interpretation of Eq. (13) using renewal theory does, however, confirm those in [1].

4. CONCLUSION

In this paper, we have developed and illustrated a new approach for the stability and throughput analysis of VT-CSMA. Our principal contributions are the two Markov chain models in Section 1; and the stability inequalities, Eqs. (9) and (10) which are exact, Eq. (11) which is approximate, the Proposition from which the throughput relation, Eq. (13), follows, and the connection between the latter and Eq. (17) which is a consequence of renewal theory, all in Section 3. It was found that Eq. (11) is useful in stability tradeoff studies over the channel parameter set $(\lambda, T, R, \alpha, \eta)$, particularly in determining regions of stable operation and in establishing the robustness of VT-CSMA stability in the presence of channel parameter uncertainties. The throughput analysis served to demonstrate the range of throughputs achievable by VT-CSMA with channel stability guaranteed.

The approximation of Eqs. (9) and (10) by Eq. (11) was necessary in order to obtain a tractable stability relation. The approximation of $(1-\alpha/j)^j$ by $e^{-\alpha}$ for $j \geq 1$ and $\alpha < < 1$ can be improved to any desired degree, but at the expense of increasing the number of inequalities which must be satisfied. Specifically, Eq. (9) may be evaluated for each $j = 1, 2, \cdots, J$ where J is chosen to attain the desired accuracy. The above approximation is then used for all $j \geq J + 1$. Coupled with Eq. (10) which is independent of j, one then has a system of $J + 2$ inequalities to consider. We recall that our other approximation, that of setting $w = \min(\eta,D) = \eta$, is justified by modifying the protocol so that the window on the virtual time axis moves only when $D \geq \eta$. This can certainly be implemented, but further study of the implications of this assumption is warranted.

The retransmission policy $f = \alpha/j$, $j \geq 1$, requires that each station monitor the channel to determine the current number of blocked stations. The underlying problem is one of either developing an algorithm which estimates j as a function of observed channel activity or of approximating $f = \alpha/j$ via a suitable backoff algorithm, also as a function of observed channel activity. This is clearly an area of worthwhile future research from both the theoretical and practical viewpoints.

REFERENCES

1. M. L. Molle, Unifications and Extensions of the Multiple Access Communications Problem, Ph.D. thesis, Computer Sci. Dept., Univ. of Calif., Los Angeles, Calif., July 1981. (Also Rept. No. CSD-810730).

2. A. G. Pakes, "Some conditions for ergodicity and recurrence of Markov chains," Oper. Res., vol. 17, pp. 1058-1061, 1969.

3. J. S. Meditch and C.-T. A. Lea, "Stability and optimization of the CSMA and CSMA/CD channels," IEEE Trans. Commun., vol. COM-31, no. 6, pp. 763-774, June 1983.

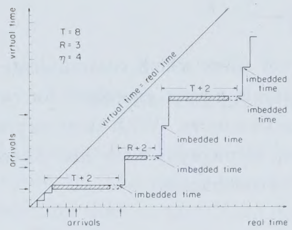

virtual time
arrivals
T = 8
R = 3
η = 4
virtual time = real time
T + 2
R + 2
T + 2
imbedded time
imbedded time
imbedded time
imbedded time
imbedded time
imbedded time
arrivals
real time

Fig. 1

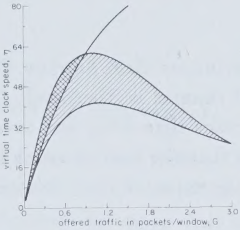


Fig. 5

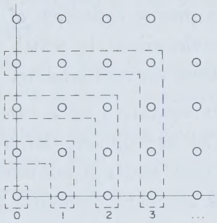


Fig. 2

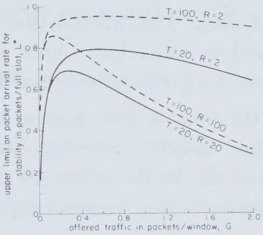


Fig. 6

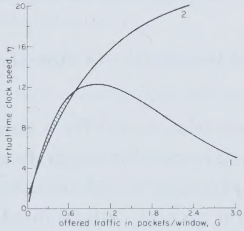


Fig. 3

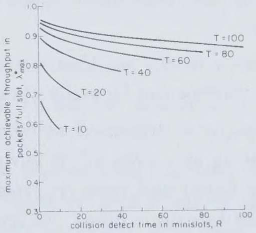


Fig. 7

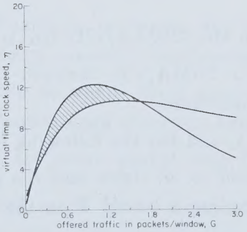


Fig. 4

Optimal Flow Control in a CSMA/CD Environment

Edward J. Coyle and Aurel A. Lazar

Departments of Electrical Engineering
Purdue University and Columbia University

Abstract

A continuous time Markov Chain model of a pair of users which communicate via a CSMA/CD channel is developed. This model provides simple matrix equations for computing the average time delay and average throughput of the two users. With these expressions it is shown that the control which maximizes the expected throughput while not exceeding a bound on the expected time delay is a window flow control mechanism.

I. Introduction

The problem of optimal flow control in local area networks has evaded solution for quite some time. The difficulty has been the lack of both a precise definition of the control problem and an exact but simple model of the network environment in which the control must operate. For the general distributed network, these pieces are still missing.

For the case of flow control between a pair of users operating in a random access environment, however, they have recently been found. The appropriate definition of the control problem -- maximization of the expected throughput under a bounded expected delay constraint -- has been developed in [3, 5]; the network modeling technique for random access networks -- which produces a simple matrix representation of the network -- can be found in [1, 2]. In this paper, we combine these two results to derive the optimal flow control between two users which communicate via a single CSMA/CD channel they share with other users.

The main results are that a window type flow control is optimal and that the parameters of the window can be easily obtained from the model.

This paper is organized as follows. In section II a queueing model for a pair of users communicating in a CSMA/CD environment is presented. The equilibrium probabilities are analytically found and exact expressions for the average throughput and the average time delay for the two users given. In section III the flow control problem is formalized and cast in terms of an optimization problem. This optimization problem is subsequently solved by linear programming methods.

II. A Queueing Model for Two Users Interacting in the CSMA/CD Environment

Two users are assumed to be communicating via a CSMA/CD channel. They are modeled as two queues interacting in a random access environment.

The queueing system we consider is shown in Figure 1, and has the following features:

(i) Queue A and B have exactly N packets between them at all times and A's goal is to maximize the throughput of queue B while not exceeding a bound T on the expected time a packet spends in queue B.

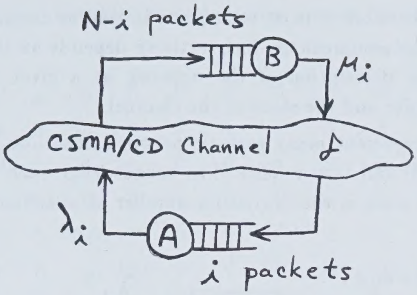


Figure 1: Two users communicating in a CSMA/CD environment.

(ii) If there are i packets in A's buffer, then A attempts to access the CSMA/CD channel at rate λ_i. This rate is assumed to be less than c, which is the inverse of the mean idle detection time on the CSMA/CD channel. This is the absolute upper limit on the rate at which the user may access the channel. Any other considerations, such as limiting the amount of channel capacity A may access, would result in a lower value of c. The optimal selection of the access rates λ_i $i = 1,...,N$, given the bound c, is the subject of this paper.

(iii) The rate at which B attempts to access the CSMA/CD channel when i packets are in its buffer is μ_i and is assumed to be fixed.

(iv) The channel operates under the nonpersistent asynchronous CSMA/CD protocol [2]. The channel load is generated both by A and B at the rates defined above and by a large group of other users These other users access the channel at a total rate δ.

The operation of the CSMA/CD channel the users share is described by the state transition diagram in Figure 2. Each block in the diagram is a distinct state for the CSMA/CD channel, which also provides the rate at which each transition between states takes place. The channel states are:

0 The 0 state is the idle state, no users transmitting on the channel;

1 the 1 state occurs when one user is transmitting but has not yet been heard by all others;

D the D, or departed state, represents a packet whose transmission has been completed before the detection time has expired. While in this state the packet can be thought of as flying down the channel. It corresponds to very short packets in most cases.

P the P state is the protected state in which the user's transmission cannot be interfered with;

C and, finally, C represents the collision state of the channel which occurs when two or more users transmit simultaneously.

The rate at which the 0 state is left depends on the channel load the users generate; the mean transmission detection time is $1/\alpha$; the mean time to the truncation of a collision is $1/\beta$; and the mean packet lengths are $\frac{1}{\nu_A}$ for user A, $\frac{1}{\nu_B}$ for user B, and $\frac{1}{\nu_C}$ for the other network users. The time spent in each state is therefore exponentially distributed; for generalizations to other distributions see [2]. In short, the channel is modeled as a continuous time Markov Chain.

The channel state the network is in at each point in time is determined by the activity of the users. As stated in the assumptions, their activity depends on the number of packets in their buffer. A complete description of the network at a given time consists of the number of packets in A's buffer and the state of the channel.

The above model can represent many realistic network situations. For instance, user A may be sending packets to B and may require B to acknowledge each packet. The limit of N packets between the two users is the maximum number of unacknowledged packets user A is allowed to have out.

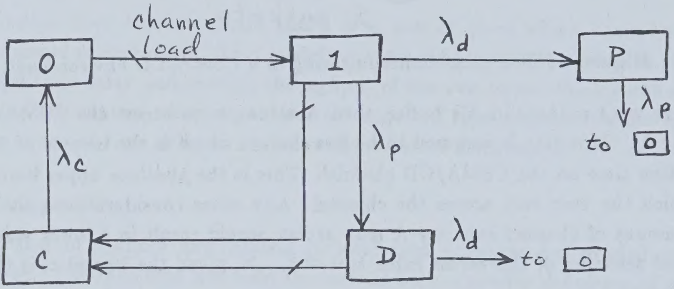


Fig. 2 The state transition diagram of the CSMA/CD channel

We now develop the expressions for the expected throughput and the expected delay for the above model. Once these have been obtained, it is possible to determine the optimal flow control policy.

To keep track of the interaction of users A and B and the network, the states shown in Figure 2 must be modified. The modification required is the splitting of the 1, D, P and C states. With this splitting the set of possible states when there are i packets in user A's buffer becomes

$$
\begin{aligned}
&[\,Q_C, Q_A, Q_B\,]_i \qquad (1)\\
&= [\;000 \quad 100\;010\;001 \quad D00\;0D0\;00D \quad P00\;0P0\;00P \quad C00\;CC0\;C0C\;0CC\;CCC\;]_i
\end{aligned}
$$

The first index in each triple represents the activity of the network on the channel. The second index in each triple is the state of user A's interaction with the channel, while the third is the state of user B's interaction. For example, $[0CC]_i$ represents a collision on the channel between a packet from user A and one from user B when there are i packets in A's buffer.

As a demonstration of the activity of this system, $[\,000 \to 001 \to 00P \to 000\,]_i$ represents a successful transmission by user B. The sequence $[\,000 \to 010 \to CC0 \to 000\,]_i$ represents a packet transmitted by user A, which is then hit by a packet from one of the users in the network, throwing the channel into the collision state. After the collision is cleared, the channel returns to the idle state.

The vector of states given in equation (1) accounts only for those states when i packets are in user A's buffer. The state space for the entire system is the union of $[\,Q_C, Q_A, Q_B\,]_i$ for $i = 0,1,\ldots,N$.

Let the matrix A(i) represent the interaction among the states in $[Q_C, Q_A, Q_B]_i$, then

$$
A(i) = \begin{bmatrix}
\mathrm{000} & \delta & \lambda_i & \mu_i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \mathrm{100} & 0 & 0 & \nu_C & 0 & 0 & \alpha & 0 & 0 & \delta & \lambda_i & \mu_i & 0 & 0 \\
0 & 0 & \mathrm{010} & 0 & 0 & \nu_A & 0 & 0 & \alpha & 0 & 0 & \delta & 0 & \mu_i & 0 \\
0 & 0 & 0 & \mathrm{001} & 0 & 0 & \nu_B & 0 & 0 & \alpha & 0 & 0 & \delta & \lambda_i & 0 \\
\alpha & 0 & 0 & 0 & \mathrm{D00} & 0 & 0 & 0 & 0 & 0 & \delta & \lambda_i & \mu_i & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \mathrm{0D0} & 0 & 0 & 0 & 0 & 0 & \delta & 0 & \mu_i & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \mathrm{00D} & 0 & 0 & 0 & 0 & 0 & \delta & \lambda_i & 0 \\
\nu_C & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{P00} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{0P0} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{00P} & 0 & 0 & 0 & 0 & 0 \\
\beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{C00} & 0 & 0 & 0 & 0 \\
\beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{CC0} & 0 & 0 & \mu_i \\
\beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{C0C} & 0 & \lambda_i \\
\beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{0CC} & \delta \\
\beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{CCC}
\end{bmatrix}
$$

The interaction between the states at level i, when i packets are in A's buffer, and levels i-1 and i+1 consists of only four transitions. There are two transitions from level i to level i-1, each of which represents a packet transfer from user A to user B. The first is the departure of a packet that is longer than the detection time, $[\,0P0\,]_i \rightarrow [\,000\,]_{i-1}$ and occurs at rate ν_A; the second is $[\,0D0\,]_i \rightarrow [\,000\,]$, representing the departure of a packet that was shorter than the detection time and occurring at rate α. The two transition from level i to level i+1, are by analogy to the cases just treated, $[\,00P\,]_i \rightarrow [\,000\,]_{i+1}$ at rate ν_B and $[\,00D\,]_i \rightarrow [\,000\,]_{i+1}$ at rate α. The matrices representing these two transfers have only two nonzero entry each. By abuse of notation we define

$$\nu_A = [0\;0\;0\;0\;0\;\alpha\;0\;0\;\nu_A\;0\;0\;0\;0\;0\;0]^T \tag{3}$$

$$\nu_B = [0\;0\;0\;0\;0\;0\;\alpha\;0\;0\;\nu_B\;0\;0\;0\;0\;0]^T$$

$$1_{000} = [1\;0\;0\;\cdots\;0] \tag{4}$$

The matrices representing these transfers are then $\nu_A 1_{000}$ and $\nu_B 1_{000}$.

The generator matrix for the continuous time Markov Chain with state space $\bigcup_{i=0}^{N} [\,Q_C, Q_A, Q_B\,]_i$ is tridiagonal, where the matrices on the diagonals were defined above. For instance, the rows of the generator corresponding to the states in which i packets are in A's buffer are

$$\begin{bmatrix} 0 & 0 & \cdots & 0 & \nu_A 1_{000} & A(i) & \nu_B 1_{000} & 0 & \cdots & 0 & 0 \end{bmatrix} \tag{5}$$

where the A(i) matrix is on the diagonal and the zeros are interpreted as null matrices of appropriate dimensions. Each entry on the main diagonal of each matrix A(i) in equation (5) is, of course, replaced with the negative of the sum of all the other elements in the corresponding row. We also note here that each matrix A(i) is invertible.

Some boundary conditions must be accounted for. The matrices A(0) and A(N) differ from the other A(i) in that the states corresponding to activity of user A and user B, respectively, are removed. Thus, for A(0), in which there are no packets in A's buffer, the rows and columns in A(0) corresponding to the states 010, 0D0, 0P0, CC0, 0CC, and CCC are deleted. The corresponding rows in the matrix $\nu_B 1_{000}$ are also deleted (so it is no longer square). Symmetric comments apply to A(N) with user B.

With the generator well defined it is possible to find the equilibrium probabilities of the chain. For each state in $[\, Q_C, Q_A, Q_B \,]_i$, define $p_i(\, Q_C, Q_A, Q_B \,)$ to be its limiting probability and P(i) to be the vector of these limiting probabilities when i packets are in A's buffer. The following recursion provides these probabilities

$$P(i) = P(i-1)R(i) \tag{6}$$

where the matrix R(i) is computed recursively from

$$R(N) = -\nu_B 1_{000} A^{-1}(N) \tag{7}$$

$$R(i) = -\nu_B 1_{000}\Big[A(i) + R(i+1)\nu_A 1_{000}\Big]^{-1} \quad , \tag{8}$$

for all $i, 0 \leq i \leq N-1$.

Remark: When the matrices A(i) are independent of i, the recursion for the P(i) takes on a very simple form; namely R(i) = R for i=2,...,N-1 where R is the solution [5] to

$$R^2 \nu_A 1_{000} + RA + \nu_B 1_{000} = 0 \quad \text{and} \quad R\nu_A 1_{000} e = \nu_B 1_{000} e \tag{9}$$

and is given by

$$R = -\nu_B 1_{000}\Big[A + \nu_B 1_{000}\Big]^{-1} \tag{10}$$

The matrices R(1) and R(N) must still be computed as above. The matrix R appears when writing down the final throughput and delay results.

Once the limiting probabilities P(i) have been determined, it is possible to determine the throughput for each user and the network, and to determine the time delay experienced by a packet in user B's buffer. Since the equilibrium probability that i packets are in A's buffer is

$$P(i)[1 \;1 \;1 \;\cdots\; 1\,]^T \tag{11}$$

the expected number of packets in B's buffer is given by

$$N - \sum_{i=1}^{N} iP(i)[1 \;1 \;1 \;\cdots\; 1\,]^T \quad . \tag{12}$$

The throughput E γ_N for user B is given by

$$E\ \gamma_N = \sum_{i=0}^{N-1} P(i)[0\ 0\ 0\ 0\ 0\ 0\ \alpha\ 0\ 0\ \nu_B\ 0\ 0\ 0\ 0\ 0]^T \tag{13}$$

where the transpose matrix picks out the states in level i in which a successful transmission takes place. The throughput for the network, excluding users A and B is given by

$$\sum_{i=0}^{N} P(i)[0\ 0\ 0\ 0\ \alpha\ 0\ 0\ \nu_C\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^T \tag{14}$$

Another important value to compute is the expected time delay through user B, which we call E τ_N. It is easily found, using Little's result, to be

$$E\ \tau_N = \{N - \sum_{i=1}^{N} iP(i)[1\ 1\ \cdots\ 1\ ]^T\}/E\gamma_N\ . \tag{15}$$

III. The Optimal Control

In the sequel it is shown that by using a maximum average throughput under a bounded average time delay criterion, the optimal flow control of two users interacting via a CSMA/CD channel is a window flow control mechanism. The window size can be obtained from the parameters of the CSMA/CD protocol introduced above. Our proof is based on a linear programming approach [4], [7]. Following [2], [3] a series of definitions will be first given.

Definition 1. $\lambda=(\lambda_i)$, $1\leq i\leq N$, will hereafter denote the control.

Definition 2. The class of controls $\lambda=(\lambda_i)$, $1\leq i\leq N$, satisfying the peak constraint

$$0\leq\lambda_i\leq c\quad ,$$

for all i, $1\leq i\leq N$, where c, $c\in R_+$, is a constant is called admissible.

Definition 3. The control $\lambda=(\lambda_i)$, $1\leq i\leq N$, is said to be optimum over the class of admissible controls for a given T, $T\epsilon R_+$ if the maximum

$$\max_{E\ \tau_N\leq T} E\ \gamma_N \tag{16}$$

is achieved.

Definition 4. The mapping $F:R_+\rightarrow R_+$ given by

$$F(T)=\max_{E\ \tau_N\leq T} E\ \gamma_N \tag{17}$$

is called the throughput time delay function.

The optimal control λ is the solution to the optimization problem defined by equation (16). Lemma 1 bellow gives a precise mathematical formulation to this problem.

Lemma 1. $\lambda=(\lambda_i), 1\leq i\leq N$, *is optimum in the class of admissible controls for a given* T, $T \in R_+$, *if it achieves the maximum of*

$$\sum_{i=0}^{N-1} P(i)[\,0\;0\;0\;0\;0\;0\alpha\;0\;0\;\nu_B\;0\;0\;0\;0\;0\,]^T \tag{18}$$

subject to the constraints $P(i) = P(i-1)R(i)$, *for all* i, $1\leq i\leq N$, (19)

$$\sum_{i=0}^{N} P(i)=1 \quad , \tag{20}$$

and

$$\sum_{i=0}^{N-1} P(i)[\,N-i\;\; N-i \cdots (N-i)-T\alpha \cdots (N-i)-T\nu_B \cdots N-i\,]^T \leq 0 \quad , \tag{21}$$

with $\lambda_i \leq c$ *for all* i, $1\leq i\leq N$.

Proof: Equation (18) specifies the throughput already arrived at in equation (13) above. Inequality (21) is the result of the time delay constraint $E\tau_N \leq T$ combined with equation (15). Equations (19) and (20) describe the constraints on the probabilities of the Markovian system under consideration. □

In what follows we will show that the above optimization problem can be reduced to a linear programming problem. Let the matrix B(i), i=0,1, $\cdots$,N, be defined as

$$B(i) = \begin{bmatrix}
-(\delta+\mu_i) & \delta & c & \mu_i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & A2 & 0 & 0 & \nu_C+c & 0 & 0 & \alpha & 0 & 0 & \delta & c & \mu_i & 0 & 0 \\
A1 & \delta & 010 & \mu_i & \delta & \nu_A & \mu_i & 0 & \alpha & 0 & 0 & \delta & 0 & \mu_i & 0 \\
0 & 0 & 0 & A3 & 0 & 0 & \nu_B+c & 0 & 0 & \alpha & 0 & 0 & \delta & c & 0 \\
\alpha & c & 0 & 0 & A4 & 0 & 0 & 0 & 0 & 0 & \delta & c & \mu_i & 0 & 0 \\
0 & \delta & 0 & \mu_i & \delta & 0D0 & \mu_i & 0 & 0 & 0 & 0 & \delta & 0 & \mu_i & 0 \\
0 & 0 & 0 & c & 0 & 0 & -\delta & 0 & 0 & 0 & 0 & 0 & \delta & c & 0 \\
\nu_C & 0 & 0 & 0 & 0 & 0 & 0 & P00 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0P0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 00P & 0 & 0 & 0 & 0 & 0 \\
\beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & C00 & 0 & 0 & 0 & 0 \\
\beta & -(\beta+\mu_i) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & CC0 & \mu_i & 0 & \mu_i \\
\beta & 0 & 0 & 0 & -(\beta+\mu_i) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\beta & 0 & c \\
\beta & 0 & 0 & -(\beta+\delta) & 0 & 0 & -(\beta+\delta) & 0 & 0 & 0 & 0 & 0 & \delta & 0CC & \delta \\
\beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & CCC
\end{bmatrix}$$

in which $A1 = -(\nu_A+\alpha+\delta+\mu_i)$, $A2 = -(\nu_C+\alpha+\delta+\mu_i)$, $A3 = -(nu_B+\alpha+\delta)$, and $A4 = -(\alpha+\delta+\mu_i)$

and $Z(i)$ $0 \leq i \leq N$ be a $[1 \times 15]$ row matrix defined by

$$Z(i) = [\, 0 \; z_2 \; z_3 \; z_4 \; z_5 \; 0 \; z_7 \; 0 \; 0 \; 0 \; 0 \; z_{12} \; 0 \; z_{14} \; z_{15} \,] \tag{23}$$

where only the elements z_2, z_3, z_4, z_5, z_7, z_{12}, z_{14} and z_{15} are possibly nonzero.

Lemma 2. *The optimal admissible control* λ_i, $1 \leq i \leq N$, *verifies the linear system of equations:*

$$P(i-1)\nu_B 1_{000} + P(i)A(i) + P(i+1)\nu_A 1_{000} = 0 \quad , \tag{24}$$

where the $P(i)$, $0 \leq i \leq N$, *are the solution to the linear programming problem of maximizing the linear objective function*

$$\sum_{i=0}^{N-1} P(i)[\, 0 \; 0 \; 0 \; 0 \; 0 \; \alpha \; 0 \; 0 \; \nu_B \; 0 \; 0 \; 0 \; 0 \; 0 \,]^T \tag{25}$$

subject to the constraints:

$$\nu_B 1_{000} + P(i)B(i) + P(i+1)\nu_A 1_{000} - Z(i) = 0 \quad , \tag{26}$$

for all i, $0 \leq i \leq N$,

$$\sum_{i=0}^{N} P(i) = 1 \quad , \tag{27}$$

and

$$\sum_{i=0}^{N-1} P(i)[\, N-i \;\; N-i \cdots (N-i)-T\alpha \cdots (N-i)-T\nu_B \cdots N-i \,]^T + Y = 0 \quad . \tag{28}$$

with P_i, $Z(i) \geq 0$ *for all* i, $0 \leq i \leq N$ *and* $Y \geq 0$.

Proof: The matrix $A(i)$ has the property that the control λ_i appears at most two times in any row. By simple additions of the corresponding columns containing the control, i.e., columns 1 and 3, 2 and 12, 4 and 14, 5 and 12, 7 and 14 and finally 13 and 15, and by setting $\lambda_i = c$ for all i, $1 \leq i \leq N$ we obtain the transformed balance equation

$$P(i-1)\nu_B 1_{000} + P(i)B(i) + P(i+1)\nu_A 1_{000} \leq 0 \quad . \tag{29}$$

for all i, $0 \leq i \leq N$. Equation (26) can than be easily obtained from the above by introducing the slack variables $Z(i)$, $0 \leq i \leq N$. Since

$$Z(i) = P(i)[B(i) - A(i)] \quad , \tag{30}$$

for all i, $0 \leq i \leq N$, the positivity of the slack variables $Z(i)$ insures that the conditions $\lambda_i <= c$ are satisfied for all i, $0 \leq i \leq N$. The slack variable $Y \geq 0$ guarantees that the time delay constraint is also verified.

Proposition 1. *The optimal control is given by*

$$\lambda_i = \begin{cases} c & N-L+1 \leq i \leq N,\ i \neq k \\ \lambda_k & i=k \\ 0 & 1 \leq i \leq N-L \end{cases}, \tag{31}$$

for some k, $k > N-L$.

Proof: If for any feasible solution of the linear program $P(N-L-1)=0$ then $P(i)=0$ for all i, $i \leq N-L-1$. Let $(P(N-L), P(N-L+1), \cdots, P(N), Z(N-L), Z(N-L+1), \cdots, Z(N))$ be part of a basic feasible solution. Thus $Z(N-L-1) > 0$ for this solution. The variables P(i), Z(i) and Y appear only in the last $L+1$ equations of (26) and the constraints (27) and (28). Therefore only $L+3$ of them at most are positive and so at most one of the Z(i) can be positive. This proves (31). □

IV. Conclusions

The optimal flow control in a CSMA/CD environment is a bang-bang control. In hindsight, this result may not seem surprising since a window flow control mechanism is the optimal control in the absence of the CSMA/CD channel. This paper verifies that this remains true even when the users must compete with each other for access to a random channel.

Acknowledgement

The research reported here was supported in part by the National Science Foundation under Grants ECS83-07140 and ECS82-06407, and in part by the Army Research Office under Contract DAAG29-83-K-0021.

References

[1] Coyle, E. and Liu, B., "Finite Population CSMA/CD Networks", *IEEE Trans. on Comm.*, to appear.

[2] Coyle, E. and Liu, B., "A Matrix Representation of CSMA/CD Networks", *IEEE Trans. on Comm.*, to appear.

[3] Lazar, A.A., "Centralized Optimal Control of a Jacksonian Network", *Proceedings of the Sixteenth Conference on Information Sciences and Systems,* Princeton University, Princeton N.J., March 17-19, 1982.

[4] Luenberger, D.G., *Introduction to Linear and Nonlinear Programming,* Addison Wesley, Reading, MA, 1973.

[5] Neuts, M.F., *Matrix Geometric Solutions in Stochastic Models,* Johns Hopkins Univ. Press, Baltimore, MD, 1981.

[6] Robertazzi, T.G., and Lazar, A.A., "On the Modeling and Flow Control of the Jacksonian Network", *Proceedings of the Ninth International Symposium on Computer Performance Modeling, Measurement, and Evaluation* (Performance'83), Univ. of Maryland, College Park, May 25-27, 1983.

[7] Weiss, G., *Personal Communication,* April 1982.

On the Application of Linear Programming to Optimal Flow Control

Aurel A. Lazar and Thomas G. Robertazzi

Departments of Electrical Engineering,
Columbia University and SUNY at Stony Brook

Abstract

The use of linear programming to analyze and determine optimal flow control policies for queueing networks is investigated. Linear programming can be used to determine the functional relationship between the maximum average throughput and the average time delay and to determine optimal flow control policies. Practical aspects of implementing the linear programs are discussed. Several examples are also presented.

I. Introduction

Over the past several years optimal flow control strategies for computer communication systems have been implemented. Most of these strategies regulate the flow of messages on the data link or on the network level. Queueing theory emerged as the natural analytical tool for modeling the computer message flows and delays. The study of flows in queueing systems provides a firm mathematical foundation for the design and understanding of practical flow control schemes.

Consider some representative problems which have appeared in the literature. Ephremides, Varaiya and Walrand [1] proposed a model where system arrivals must be routed to one of two M/M/1 queues so as to minimize the expected total time for the completion of service on all jobs which arrive before a given time T. Rosberg, Varaiya and Walrand [15] studied a model consisting of two tandem queues. Given the number of items in both queues, the optimal bounded state dependent service rates for the first queue is sought that minimizes an expected total discounted or an average cost.

Previous work by one of the co-authors has established the optimum flow control policy for certain fundamental queueing models [8], [9]. Optimality was defined as achieving the maximum expected throughput while the expected average time delay is constrained by an upper bound. The analytical solution for extensions of the original model has been also obtained [10], [2]. The optimal control has nominally taken the form of a window type flow control mechanism. Unfortunately, the tractability of such queueing system problems depends on establishing reasonably simple expressions for the equilibrium probabilities as a function of the control. For models of practical interest it may not be possible to establish such expressions.

A mathematical programming approach to optimal flow control of queueing systems is of much practical interest. Weiss [17] has shown that linear programming can be used to solve for the optimal control of the cyclic network investigated by Lazar [8]. In this paper it is shown that the linear programming approach can also be applied for an extended model with a two, rather than one dimensional state space.

This paper is organized as follows. Section II describes a queueing model for which an optimal flow control strategy has been previously found analytically. Variations of this model which are not analytically tractable are also mentioned. In section III the flow control problem is formulated. In section IV it is shown how to convert this to a linear programming problem. Practical aspects of writing the linear program are discussed.

II. The Queueing System Model

The flow control model which will be examined consists of two tandem queues. Two tandem queue models have been previously investigated in [4], [5], [6], [7], [12], [13]. Arrivals to the first or control queue form a Poisson stream of rate δ. The first queue's service rate is λ_k, with $0 \le \lambda_k \le c$. The second queue has a service rate $\mu=(\mu_k)$, $1 \le k \le N$. There are N tokens associated with this queueing system. An arriving packet is first paired with a free token before it proceeds through the system. After leaving the second queue the token proceeds back to the first queue by means of an acknowledgment packet. If no free tokens are available at the control queue to pair with an arrival, the arrival is blocked. The latter is assumed to then proceed by a different route or is discarded.

This flow control model is illustrated in Figure 1 below:

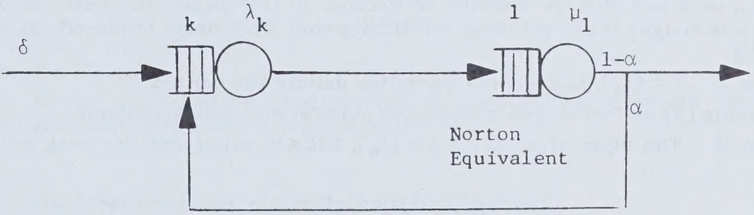


Fig.1 An open-closed model for optimal flow control.

The model also includes a mechanism for accounting for transmission errors. After being received, packets are found to be unintelligible with independent probability α. This causes a request for a retransmission from the control node.

The equilibrium probabilities of having a pair of packets (k,l) in the queueing system of Fig. 1 are given by [14]:

$$p_{kl} = \prod_{i=1}^{l} \frac{\delta}{(1-\alpha)\mu_i} \prod_{j=1}^{k} \frac{\delta}{(1-\alpha)\lambda_j} p_{00} \quad , \tag{1}$$

for all k, l, $k+l \leq N$. This is a very appealing result since as Konheim and Reiser point out there are few solutions to queueing systems with blocking in the literature [4].

A natural extension of this model is to allow packets to enter the first queue without regard to the number of free tokens present. An upper limit is still placed on the number of packets in the first queue. Unfortunately this leads to expressions for the equilibrium probabilities which are much less tractable. This provides a motivation for investigating the linear programming approach (see also section IV). Linear programming can also solve problems in which μ is not a concave function. These cases are not easily solved analytically [14].

III. The Optimization Problem

The average throughput and the average time delay of the second or receiving queueing system are given by [9], [10]:

$$E\gamma_N = \sum_{l=1}^{N} \mu_l p_{\cdot l}$$

and

$$E\tau_N = \frac{\sum_{l=1}^{N} l p_{\cdot l}}{\sum_{l=1}^{N} \mu_l p_{\cdot l}} \quad ,$$

respectively, where $p_{\cdot l}$ denotes the probability of there being l packets in the receiving queue and N is the number of tokens. In this paper, the best choice of the λ_k's is sought that optimizes the throughput time delay trade-off. More specifically:

Definition 1. $\lambda = (\lambda_k)$, $1 \leq k \leq N$, will hereafter denote the control.

Following [9] we formally introduce the class of admissible controls.

Definition 2. The class of controls $\lambda = (\lambda_k)$, $1 \leq k \leq N$, satisfying the peak constraint

$$0 \leq \lambda_k \leq c \quad ,$$

for all k, $1 \leq k \leq N$, where c, $c \varepsilon R_+$, is a constant is called admissible.

Definition 3. The control $\lambda = (\lambda_k)$, $1 \leq k \leq N$, is said to be optimum over the class of admissible controls for a given T, $T \varepsilon R_+$, if the maximum

$$\max_{E\tau_N \leq T} E\gamma_N$$

is achieved.

Definition 4. The mapping $F: R_+ \to R_+$ given by

$$F(T) = \max_{E\tau_N \leq T} E\gamma_N$$

is called the throughput time delay function.

Based on the latter definition we arrive at the following mathematical optimization problem:

$$\max \sum_{l=1}^{N} \mu_l p_{\cdot l} \tag{2}$$

under the constraints

$$\frac{\sum_{l=1}^{N} l p_{\cdot l}}{\sum_{l=1}^{N} \mu_l p_{\cdot l}} \le T \tag{3}$$

and

$$\sum_{l=0}^{N} p_{\cdot l} = 1 \ . \tag{4}$$

Additional constraints are provided by the global balance equations [3]:

$$A(\delta, \lambda_k, \mu_l) \cdot p = 0 \quad . \tag{5}$$

where δ is the input Poissonian process rate, λ_k is the state dependent controller queue service rate, μ_l is the state dependent controlled queue service rate, $p_{\cdot l}$ is the equilibrium probability of the l'th state, and

$$p_{\cdot l} \ge 0, \tag{6}$$

for all l, $1 \le l \le N$ and

$$0 \le \lambda_k \le c \quad , \tag{7}$$

for all k, $1 \le k \le N$.

In the mathematical programming problem above, although the $p_{\cdot l}$'s are naturally unknowns, these depend on the λ_k's.

IV. The LP Formulation of the Optimization Problem

If $\delta \to \infty$ in the above queueing model, a cyclic queueing system with two queues and N circulating tokens results. Lazar originally proposed the criterion of maximizing the expected throughput subject to an upper bound on the expected time delay in the context of this model [8]. Subsequently it has been shown by Weiss that this optimization problem can be formulated in terms of a linear program [17].

Weiss developed the linear programming approach of the one dimensional Markov state model by using the detailed balance equations [3]:

$$\mu_l p_{\cdot l} - \lambda_l p_{\cdot l-1} = 0 \quad ,$$

for all l, $1 \le l \le N$. To produce a solvable linear program, he replaced λ_l by its bound c and appended the slack variable z_l to obtain

$$\mu_l p_{\cdot l} - c p_{\cdot l-1} + z_l = 0 \quad ,$$

for all l, $1 \le l \le N$.

The above optimization problem can then be formulated as:

$$\max \sum_{l=1}^{N} \mu_l p_{\cdot l} \tag{8}$$

under the conditions that

$$\sum_{l=1}^{N} (l - \mu_l T) p_{\cdot l} + y = 0 \quad , \tag{9}$$

$$\sum_{l=0}^{N} p_{\cdot l} = 1 \quad , \tag{10}$$

$$A(\delta, c, \mu_l) \cdot p + z = 0 \quad , \tag{11}$$

and

$$y, z, p_{\cdot l} \ge 0 \quad , \tag{12}$$

for all l, $0 \le l \le N$.

Equation (9) represents the time delay constraint with slack variable y. As already mentioned above, the λ_k's have been replaced by the upper bound c and slack variables have been appended to the balance equation (5) to obtain equality (11). The balance equations are bi-linear in the parameters and the equilibrium probabilities, respectively. Once the probabilities and slack variables are determined by means of the above linear program, the balance equations may be linearly solved for the λ_k's.

For the optimal solution, the z_l's are zero except for the one associated with the control boundary enforced by (9) and one associated with the adjacent zero probability state. The latter variable compensates for the flow out of the boundary non-zero probability state and into the adjacent zero probability state.

If the control policy is known, the expected throughput and time delay can be determined by solving the linear set of global balance equations for the equilibrium probabilities. Numerically stable and computationally efficient techniques can be used [16].

The above method provides a workable computational tool for obtaining both equilibrium probabilities and the optimal control for a variety of problems which are not analytically tractable. Moreover it allows the formulation of the above optimization problem in the context of the well established body of linear programming theory [11]. Among linear objective functions of practical interest are:

$$\sum_{l=1}^{N} a_l p_{\cdot l} \tag{13}$$

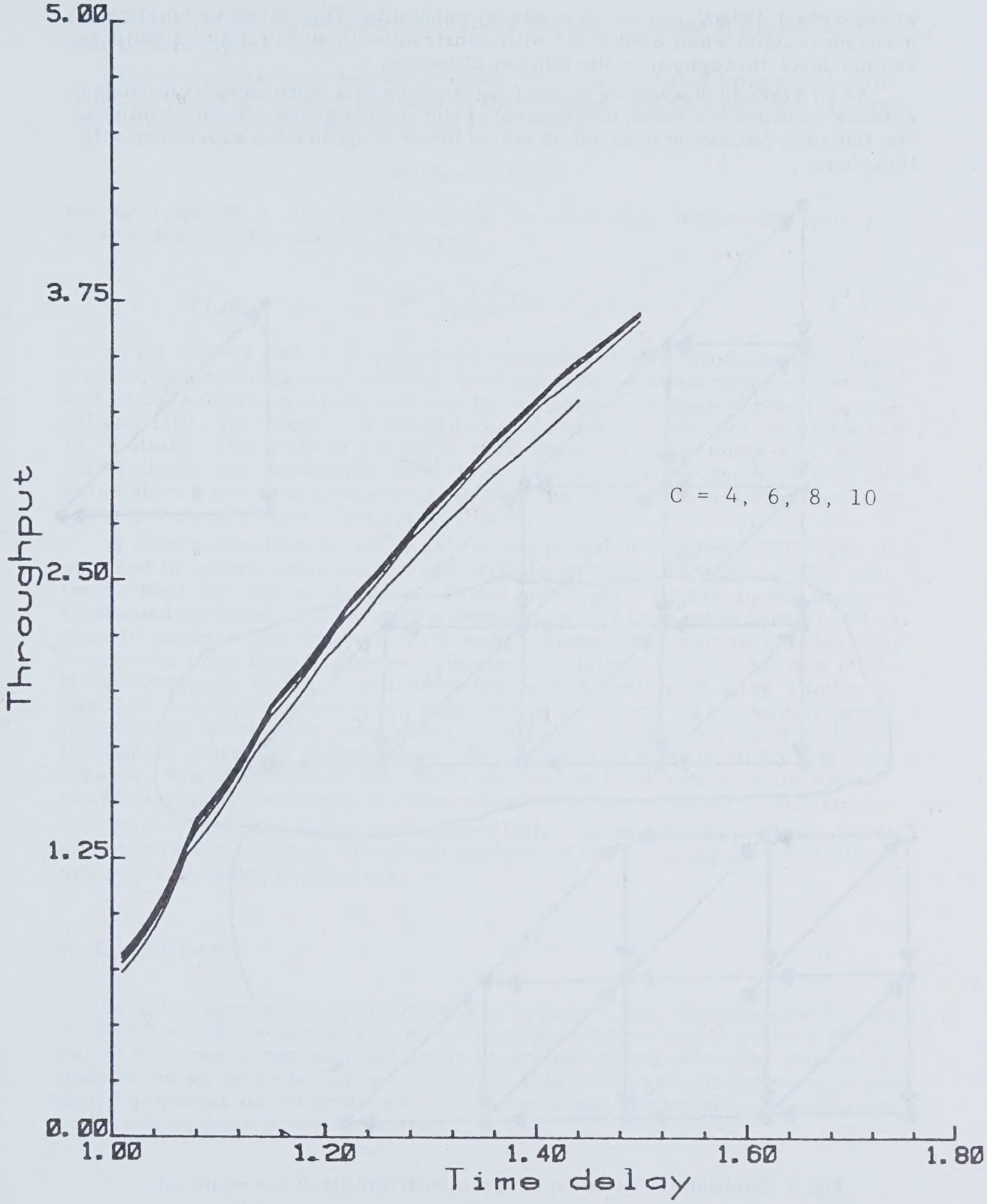
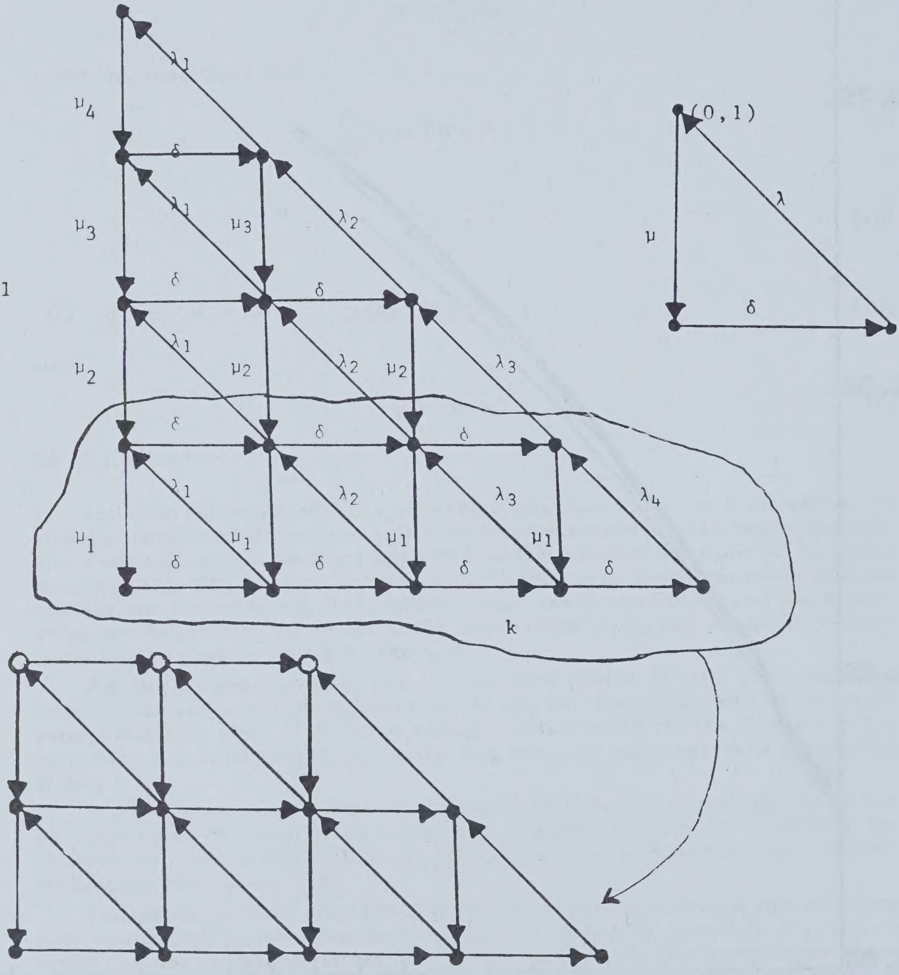


Fig.2 The Throughput Time Delay Function (δ =5)

where $a = (a_l)$, $1 \le l \le N$, is a set of arbitrary constants. This objective function is made meaningful when associated with constraints on such related quantities as time delay, throughput or the number of tokens.

As an example of applying linear programming to a more complex queueing network consider the token flow control of the system of the previous sections. The following discussion is based on actual linear programming experience with this model.



λ_1
μ_4
(0,1)
λ
μ
δ
μ_3
μ_3
λ_2
1
μ_2
μ_2
μ_2
λ_3
λ_4
μ_1
μ_1
μ_1
μ_1
k

Fig. 3 Solution to the linear program with one slack per equation.

The linear program incorporates the state equations for a large number of tokens. If the optimal control consisted of using less than the full number of tokens, then those states associated with a too large number of tokens will have zero probability.

The state diagram is two dimensional. To obtain a linear programming formulation the information contained by the detailed balance equations concerning the equilibrium probabilities is utilized. Since

$$\mu_{l+1} p_{k,l+1} = \delta p_{k,l} \quad , \tag{14}$$

for all l, $0 \le l \le N-1$, the probability p_{kl} is calculated recursively from p_{k0}, independently of the control. We have:

$$p_{kl} = \prod_{i=1}^{l} \frac{\delta}{\mu_i} p_{k,0} \quad , \tag{15}$$

for all k, l, $0 \le k \le N$, $1 \le l \le N$. The p_{k0}'s are computed directly from the global balance equations along the boundary $\{0 \le k \le N, l=0\}$. A combination of detailed and global balance equations can now be included as LP constraints along with (9) and (10). Therefore, one needs to associate slack variables only with the $\{k, 0\}$ states. This leads to a number of slacks equal to the number of tokens. Fig. 2 shows the throughput time delay function for various values of the parameters δ and c as obtained by running the linear program. The server of the receiving queue is $\mu = (10l/(9+l))$ for all l, $1 \le l \le N$.

A different method to arrive at a linear programming formulation can be obtained by associating a single slack variable with each λ transition. The solutions which now appear, however, do not correspond to the access protocol mentioned previously. To exemplify, assume that the time delay bound is set to one. In terms of the window control, only a single token window type solution would satisfy this bound. However, a better and optimal solution consists of the states illustrated in Fig 3. It corresponds approximately to using a buffer of size 9 for the input queue and a single token for access to the output queue. This system meets the time delay constraint. It also leads to a better throughput since the probability of the output queue being empty has been reduced. It is optimal since the input buffer size and the number of tokens cannot be increased. Increasing the time delay bound leads to similar solutions.

This solution is valuable as it demonstrates that the linear program can to a limited extent, optimize the access protocol of the queueing system itself, as well as the queueing parameters.

V. Conclusions

In this paper a linear programming approach has been applied to the optimal flow control of a tandem queueing system with a blocking access protocol. It has been shown that the linear programming approach may allow solutions based on an undesired set of states. This formulation is appropriate for deriving optimal access protocols. The methodology employed is appealing for both analytical and practical reasons.

Acknowledgement

The research reported here was supported in part by the the National Science Foundation under Grant # NSF ECS-81-08776 and in part by the Army Research Office under Contract DAAG29-83-K-0021.

References

[1] Ephremides A., Varaiya, P. and Walrand, J., "A Simple Dynamic Routing Problem", *Trans. on Automatic Control*, Vol. AC-25, No. 4, Aug. 1980.

[2] Hsiao, M.T. and Lazar, A.A., "On the Data Link Flow Control with Partial Observations", *Proceedings of the Conference on Information Sciences and Systems*, Johns Hopkins University, Baltimore, March 17-19, 1983.

[3] Kelly, P., Reversibility and Stochastic Networks, Willey, 1979.

[4] Konheim, A.G. and Reiser, M., "Finite Capacity Queuing Systems with Applications in Computer Modeling", *SIAM J. Computing*, Vol. 7, No. 2, May 1978.

[5] Konheim, A.G., and Reiser M., "A Queuing Model with Finite Waiting Room and Blocking", *JACM.*, Vol. 23, No. 2, April 1976.

[6] LaTouche, G. and Neuts, M.F., "Efficient Algorithmic Solutions to Exponential Tandem Queues with Blocking", *SIAM J. of Algebraic and Disc. Methods*, Vol. 1, No.1, March 1980, pp. 93-106.

[7] Lam, S.S. and Lien, Y. L., "Modeling and Analysis of Flow Controlled Packet Switching Networks", *Proceedings of the Data Communication Symposium*, Mexico City, 1981.

[8] Lazar, A.A., "Optimal Control of a Class of Queueing Networks in Equilibrium", *IEEE Trans. on Automatic Control*, Vol. AC-28, Nov. 1983.

[9] Lazar, A.A., "Centralized Optimal Control of a Jacksonian Network", *Proceedings of the Sixteenth Conference on Information Sciences and Systems*, Princeton University, Princeton N.J., March 17-19, October 1982.

[10] Lazar, A.A., "Data Link Flow Control with Partial Observations", *Proceedings of the Twentieth Allerton Conference on Communication, Control and Computing*, University of Illinois, Urbana-Champaign, Ill., Oct. 6-8, 1982.

[11] Luenberger, D.G., *Introduction to Linear and Nonlinear Programming*, Addison Wesley, Reading, MASS, 1973.

[12] Perros, H.G., "A Two-Node Queue Network With A Maximum Number of Allowable Jobs", *Performance '83*, University of Maryland, College Park, Maryland, June 1983.

[13] Pittel, B., "Closed Exponential Networks of Queues with Saturation: The Jacksonian Stationary Distributions and its Asymptotic Analysis", *Math. of Op. Res.*, Vol. 4, No. 4, Nov. 1979.

[14] Robertazzi, T.G., and Lazar, A.A., "On the Modeling and Flow Control of the Jacksonian Network", *Proceedings of the Nineth International Symposium on Computer Performance Modelling, Measurement, and Evaluation* (Performance'83), Univ. of Maryland, College Park, May 25-27, 1983.

[15] Rosberg, Z., Varaiya, P. and Walrand, J., "Optimal Control of Service in Tandem Queues", Memorandum No. UCBN/ERL M80/42, September 1980.

[16] Stewart, G., *Introduction to Matrix Computation*, Academic Press, New York, 1973.

[17] Weiss, G., *Personal Communication*, April 1982.

The Throughput Time Delay Trade-off of a Birth-Death Process

Hamid K. Ahmad and Aurel A. Lazar

Columbia University
Department of Electrical Engineering
Computer Communications Research Laboratory

ABSTRACT

Two objective functions are used to analyze the throughput time delay trade-off of a birth-death process. The first is defined by the maximum of the average throughput subject to a bounded average time delay. The resulting optimization problem is solved by analytical as well as linear programming methods. The second, is a linear combination of the average throughput and the average time delay. This optimization problem is solved by means of dynamic programming. The relative merits of the above approaches to optimal flow control in computer communication networks are also discussed.

I. Introduction

The prevention of throughput degradation due to overload, deadlock avoidance, and fair allocation of network resources are among the desirable objectives in the design of computer communication networks. Early work on the design of such networks emphasized optimal placement of communication links, choice of channel capacities, and routing messages [8]. These problems continue to be of important practical and analytical interest.

The throughput time delay trade-off plays a central role in finding and characterizing optimal flow control policies for computer communication networks [4-6]. Without any loss of generality the first order statistics of the input-output behavior of closed Markovian queueing systems is the same as the one of the Norton equivalent birth-death process [1]. These results lead us to consider and analyze the throughput time delay trade-off of a birth-death process. The relative merits of two objective functions to optimal flow control are investigated. The mathematical programming problems associated with these functions are shown to have tractable solutions. The optimal flow control policy derived using the objective functions is a window flow control mechanism.

The organization of this paper is as follows. In section II the two objective functions are defined. Section III deals with the linear programming approach while section IV presents the analytical approach to the optimization of the first objective function. The maximization of the second objective function by means of dynamic programming is given in section V. Finally, the relative merits of the above approaches to optimal flow control in computer communication networks are discussed in section VI.

II. The Objective Functions

We consider a birth-death process with a buffer of size N as shown in Fig. 1. Such a process can be chosen to have the same first order statistics as a closed Markovian queueing network with N packets and centralized controller [6]. The lower queue represents the controller. k of the total of N packets are assumed to be waiting for service in the upper queue. The remaining N-k packets are in the feedback queue (controller).

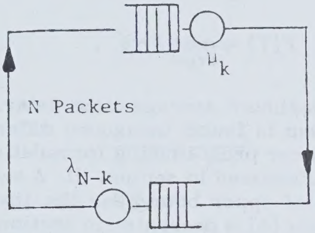


Fig.1 A birth-death process with a finite buffer.

The arrivals to the upper queueing system can be controlled by setting the service rate of the feedback queue, λ_{N-k}. Note that N, the maximum number of packets in the closed system, is an arbitrarily large integer and $\mu_0=0$ by convention. Without any loss of generality, the feedback queue models the input flow to the upper queueing system with a finite buffer size [5]. The average throughput and the average time delay of the queueing system are given by [2].

$$E\gamma_N=\sum_{k=1}^{N}\mu_k p_k$$

$$E\tau_N=\sum_{k=1}^{N}\frac{k}{\mu_k}q_k$$

respectively, where for all k, $1\le k\le N$,

$$q_k=\frac{\mu_k p_k}{\sum_{k=1}^{N}\mu_k p_k}$$

$$p_k=\prod_{l=0}^{k-1}\frac{\lambda_{N-l}}{\mu_{l+1}}p_0$$

and

$$p_0=\frac{1}{1+\sum_{k=1}^{N}\prod_{l=0}^{k-1}\frac{\lambda_{N-l}}{\mu_{l+1}}}.$$

The real numbers p_k denote the probabilities that the network (the upper queue) contains k packets ($0 \le k \le N$).

Following [4], [5] a series of definitions will first be given.

Definition 1. $\lambda=(\lambda_k)$, $1 \le k \le N$ *, will hereafter denote the control.*

Definition 2. The class of controls $\lambda=(\lambda_k)$ *is called admissible if for all* k, $1 \le k \le N$, *and constants* $c_k \varepsilon R_+$ *, it satisfies the peak constraints* $0 \le \lambda_k \le c_k$

Two objective functions are considered in this paper. The first is defined by

$$F(T) = \max_{E\tau_N \le T} E\gamma_N \quad ,$$

where T is a preassigned maximum average time delay. The solution to the resulting optimization problem is found using two different approaches. The first approach, based on a linear programming formulation in the steady state probabilities p_k, $0 \le k \le N$, is discussed in section III. A second approach, which is based on finding the actual upper bound for the throughput subject to a maximum time delay constraint [4] is presented in section IV.

The second objective function is defined as:

$$G(\alpha)=E\,\gamma_N-\alpha E\,\tau_N \quad ,$$

where $\alpha>0$ is a fixed constant. It is shown in section V that dynamic programming can be used to maximize $G(\alpha)$ over the class of all admissible controls.

III. The Linear Programming Approach

The linearity of the objective function $F(T)$ in the steady state probabilities p_k, $0 \le k \le N$, and the set of linear constraint equations in p_k can be exploited to arrive at a linear programming formulation [10]. We obtain the following linear program in canonical form [9]:

$$\max E\gamma_N = \max \sum_{k=1}^{N} \mu_k p_k$$

subject to

$$\sum_{k=0}^{N} p_k = 1,$$

$$-c_k p_{k-1} + \mu_k p_k \le 0, \tag{1}$$

for all k, $1 \le k \le N$, and

$$\sum_{k=1}^{N} (k - \mu_k T) p_k \le 0,$$

and finally

$$p_k \geq 0,$$

for all k, $0 \leq k \leq N$. The constants $c_k \varepsilon R_+$ are bounds for the range of values of the admissible controls λ_k (the service rates of the feedback queue in Fig. 1), *i.e.*, $0 \leq \lambda_k \leq c_k$, for all k, $1 \leq k \leq N$.

The control schemes and the maximum average throughput are dependent upon the service rate of the queueing network, $\mu = (\mu_k)$, $1 \leq k \leq N$. Two cases can be discussed on the basis of the following definition [6]:

Definition 3. The service rate $\mu = (\mu_k)$, $1 \leq k \leq N$, is said to be convex nondecreasing if

$$\mu_k \leq \mu_{k+1}, \tag{2}$$

and

$$\frac{\mu_k - \mu_i}{k-i} \geq \frac{\mu_j - \mu_i}{j-i} \quad , \tag{3}$$

for all i, j and k such that $0 \leq i < j < k \leq N$.

The solution of the linear program (1) can be obtained using the simplex method [7] by substituting $\lambda_k = c_k$, for all k, $1 \leq k \leq N$, in (1) and adding the slack variables y and z_k $(1 \leq k \leq N)$. Thus the equivalent linear program in the basic form is:

$$\max E\gamma_N = \max \mu' \mathbf{p}$$

subject to

$$\mathbf{A}\mathbf{p} + (\mathbf{z}', y)' = \mathbf{b} \quad , \tag{4}$$

and

$$\mathbf{p} \geq 0 \ , \quad \mathbf{z} \geq 0 \ , \quad y \geq 0 \quad ,$$

where $\mu' = (0, \mu_1, \mu_2, \cdots, \mu_N)$, $\mathbf{p}' = (p_0, p_1, \cdots, p_N)$, $\mathbf{b}' = (1,0,0, \cdots ,0)$, $\mathbf{z}' = (z_1, z_2, \cdots, z_N)$ and

$$\mathbf{A} = \left[\begin{array}{ccccccc} 1 & 1 & 1 & . & . & . & 1 \\ -c_1 & \mu_1 & 0 & . & . & . & 0 \\ 0 & -c_2 & \mu_2 & . & . & . & 0 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ 0 & 0 & 0 & . & . & -c_N & \mu_N \\ 0 & a_1 & a_2 & . & . & a_{N-1} & a_N \end{array} \right]$$

with $a_k = (k - \mu_k T)$, $1 \leq k \leq N$. A control corresponding to the optimal feasible solution of (4) is given by [10]:

$$\lambda_k = \begin{cases} 0 & \text{if } p_{k-1} = 0 \\ \mu_k p_k / p_{k-1} & \text{if } p_{k-1} > 0 \end{cases} , \tag{5}$$

for all k, $1 \leq k \leq N$. Note that the positivity of the slack variables y, z_k $(1 \leq k \leq N)$, implies that the constraints

$$c_k - \lambda_k \geq 0 \quad , \tag{6}$$

for all k, $1 \leq k \leq N$, and

$$T - E\tau_N \geq 0 \tag{7}$$

are verified.

Thus the optimal control of a birth-death process with N packets can be formulated as the linear program (4). Such a formulation enables us to compute the optimal control laws efficiently via the simplex method. The optimal control results in [10]:

Theorem 1. *The optimal control of a birth-death process with a buffer of size* N *is given by*

$$\lambda_k \equiv \begin{cases} c_k & \text{if} \quad 0 \leq k \leq L, k \neq m \\ \lambda_m & \text{if} \quad k = m, 1 \leq m \leq L \\ 0 & \text{if} \quad L < m \leq N \end{cases} . \tag{8}$$

Furthermore if $\mu = (\mu_k)$, $1 \leq k \leq N$, *is convex*

$$\lambda_k = c_k \ , \tag{9}$$

for all k, $1 \leq k \leq N$, *where the strictly positive integer* L *is given by* $T_{\max}^{(L-1)} \leq T \leq T_{\max}^{(L)}$.

The Norton equivalent μ of a network of queues is a function of the service rates of the individual queues [5]. Therefore variations in any of the individual service rates will be reflected in μ. The dual program may be used to examine the effect of small variations in the vector μ on the objective function $F(T)$. The symmetric dual for the linear program (1) is defined as

$$\min \mathbf{b}'\mathbf{x}$$

subject to

$$\mathbf{A}'\mathbf{x} \geq \mu \quad , \tag{10}$$

$$\mathbf{x} \geq 0 \quad ,$$

where $\mathbf{x}' = (x_0, x_1, \cdots, x_N)$ is the dual variable. Since (1) is also the dual to (10), we will consider the following primal-dual in the unsymmetric form [7]:

$$\min \mathbf{b}'\mathbf{x}$$

subject to

$$\mathbf{A}'\mathbf{x} = \mu \quad , \tag{11}$$

$$\mathbf{x} \geq 0 \quad .$$

$$\max \mu' \mathbf{p}$$

subject to

$$\mathbf{p}'\mathbf{A}' \leq b' \quad , \qquad (12)$$
$$\mathbf{p} \geq 0 \quad .$$

Suppose that in (11) the optimal basis is the matrix $\mathbf{B}$ with corresponding basic solution $(\mathbf{x_B}, 0)$. where $\mathbf{x_B} = \mathbf{B}^{-1}\mu$. Then the solution to the corresponding dual problem (12) is $\mathbf{p}' = \mathbf{b}_\mathbf{B}' \mathbf{B}^{-1}$.

Small changes in the vector μ will not change the optimal basis as long as nondegeneracy of the basic solution holds [7]. Thus for a change of $\Delta \mu$ the optimal solution is $\mathbf{x} = (\mathbf{x_B} + \Delta \mathbf{x_B}, 0)$ where $\Delta \mathbf{x_B} = \mathbf{B}^{-1} \Delta \mu$. Hence the corresponding increment in the objective function is given by:

$$\mathbf{b}'\Delta\mathbf{x_B} = \mathbf{p}'\Delta\mu \quad . \qquad (13)$$

Therefore $\mathbf{p}'$ gives the sensitivity of the throughput time delay function, F, with respect to small changes in the vector μ.

IV. The Analytical Approach

Explicit expression for the objective function F can be obtained analytically [5]. The state dependent service rate $\mu=(\mu_k)$, $1 \leq k \leq N$, of the birth-death process satisfies, in addition to condition (2) of Definition 3, the following inequality

$$\frac{\mu_j - \mu_i}{j-i} \geq \frac{\mu_k - \mu_i}{k-i} \quad , \qquad (14)$$

for all i,j and k such that $0 \leq i < j < k \leq N$. Thus μ is concave nondecreasing. Inequality (2) implies [5] that the throughput is increasing with the number of packets in the system. Inequality (14) on the other hand asserts that the time delay of the birth-death process increases with the number of packets in the network [5].

The solution to the optimal control of a birth-death process with a concave nondecreasing rate μ leads to an explicit expression for $F(T)$ [5]. For a given maximum time delay T, there exists an integer $L \leq N$ such that $T_{\max}^{(L-1)} < T \leq T_{\max}^{(L)}$ Let $\rho_k = \prod_{l=0}^{k-1} c / \mu_{l+1}$, $1 \leq k \leq N$, where $\rho_k \geq x_k = \prod_{l=0}^{k-1} \lambda_{N-l} / \mu_{l+1}$. Then we can state the following result [5]:

Theorem 2. *The optimal control of a birth-death process with a buffer of size of N for a given time delay T with* $T_{\max}^{(L-1)} < T \leq T_{\max}^{(L)}$, $2 \leq L \leq N$, *is given by :*

$$\lambda_k \equiv \begin{cases} c & \text{if } N-L+2 \leq k \leq N \\ \lambda_{N-L+1} & \text{if } k = N-L+1 \\ 0 & \text{if } 1 \leq k \leq N-L \end{cases} \quad , \qquad (15)$$

where

$$\lambda_{N-L+1} = \frac{1}{\frac{L}{\mu_L} - T} \sum_{l=1}^{L-1} \frac{(\mu_l T - l)\rho_l}{\rho_{L-1}} \quad . \qquad (16)$$

Furthermore, the throughput time delay function is

$$F(T)=\frac{\sum_{k=1}^{L-1}(L\mu_k-k\,\mu_L)\rho_k}{L-\mu_L T+\sum_{k=1}^{L-1}(L-k-(\mu_L-\mu_k)T)\rho_k}\,. \tag{17}$$

Remark. Since $T>T_{\max}^{(L-1)}$ it is expected that the control of the first L-1 packets will be $\lambda_k=c$ for all k, $1\le k\le L-1$. The addition of the L'th packet contributes $T-T_{\max}^{(L-1)}$ to the average time delay. Theorem 2 clearly confirms this intuitive interpretation of the control of the birth-death process, and provides explicit expressions for the optimal control law and the throughput time delay function. Note that the window size L, which depends on the line capacity into the queueing system, can be derived from the maximum time delay of the system by using the inequalities:

$$T_{\max}^{(L-1)}<T\le T_{\max}^{(L)} \tag{18}$$

Note also that equations (15) and (16) allow us to investigate the sensitivity of the optimal control law and the throughput time delay function due to perturbations of μ. Two important properties of the function $F(T)$ relevant to the sensitivity analysis are given by [5]:

Lemma 1. *The function $F:R_+\to R_+$ is continuous nondecreasing on R_+. In addition, for all T, $T_{\max}^{(L-1)}<T\le T_{\max}^{(L)}$, and $L\le N$,* F *is convex.*

V. The Dynamic Programming Approach

Let us consider now the objective function $G:R_+\to R$ defined as a linear combination of the average throughput, $E\gamma_N$, and the average time delay, $E\tau_N$:

$$G(\alpha)=E\gamma_N-\alpha E\tau_N \tag{19}$$

The fixed constant $\alpha>0$ represents the relative weight of throughput versus time delay. The objective of the controller of Fig. 1 is to maximize the function G over all admissible controls.

Using the expressions for $E\gamma_N$ and $E\tau_N$ given in section II, equation (19) can be expressed in a form appropriate for stating the optimization problem. Specifically, we have for equation (19):

$$G(\alpha)=\sum_{k=1}^{N}g(\alpha,k) \tag{20}$$

where

$$g(\alpha,k)=(\mu_k-\frac{\alpha k}{\sum_{k=1}^{N}\mu_k p_k})p_k \tag{21}$$

Let Λ denote the set of all admissible controls. Then, the optimization problem for the queueing system of Fig. 1 can be stated as:

$$\max_{\lambda\varepsilon\Lambda} G(\alpha) = \max_{\lambda\varepsilon\Lambda} \{ \sum_{k=1}^{N} g(\alpha,k) \} \tag{22}$$

where $\lambda=(\lambda_k)$, $1\leq k\leq N$.

Instead of maximizing the function G over the set Λ, however, we consider the maximization over the set P defined by all $\mathbf{p}'=(p_0,p_1,\cdots,p_N)$ such that

$$\sum_{k=0}^{N} p_k = 1,$$

$$p_k \geq 0, \tag{23}$$

for all k, $1\leq k\leq N$, and

$$p_k \leq \frac{c_k}{\mu_k} p_{k-1} \quad ,$$

where $c_k\geq 0$, for all k, $1\leq k\leq N$.

To obtain the solution to the latter optimization problem recursively we define the functions $G_m^*:R_+\to R$ by

$$G_m^*(\alpha) = \max_{p^m\varepsilon P} \{ \sum_{k=1}^{m} g(\alpha,k) \} \quad , \tag{24}$$

for all m, $2\leq m\leq N$, where $p^m = (p_k)$, $1\leq k\leq m$. Hence, from Bellman's optimality condition [3], equation (24) can be written as:

$$G_m^*(\alpha) = \max_{p_m} \max_{p^{m-1}} \{ g(\alpha,m) + \sum_{k=1}^{m-1} g(\alpha,k) \} \tag{25}$$

$$= \max_{p_m} \{ g(\alpha,m) + G_{m-1}^*(\alpha) \},$$

for all m, $2\leq m\leq N$, where, by equation (24),

$$G_{m-1}^*(\alpha) = \max_{p^{m-1}\varepsilon P} \{ \sum_{k=1}^{m-1} g(\alpha,k) \} \quad . \tag{26}$$

It is clear that for k=1

$$G_1^*(\alpha) = \max_{p_1\varepsilon P} \{ g(\alpha,1) \}. \tag{27}$$

Therefore, from the formulation of the optimization procedure for the objective function G described above, we have the following result.

Theorem 3. *The optimal control of a birth-death process with a finite buffer of size N ($N\geq 2$), and a fixed constant $\alpha\varepsilon R_+$, is a bang-bang type control policy. The corresponding objective function for the process is $G_N^*(\alpha)$.*

Proof: The last statement of the theorem follows trivially from (25). The recursive equations (25) and (27) are used to prove the first part of the theorem. From (27) we obtain p_1 satisfying $\sum_{k=1}^{N} p_k = 1-p_0$. Then, p_2 is found as the solution to

$$G_2^*(\alpha) = \max_{p_2} \{g(\alpha,2)+G_1^*(\alpha)\}$$

such that $\sum_{k=2}^{N} p_k = 1-p_0-p_1$. Similarly, the solution p_m to

$$G_m^*(\alpha) = \max_{p_m} \{g(\alpha,m)+G_{m-1}^*(\alpha)\}$$

such that $\sum_{k=m}^{N} p_k = 1-\sum_{i=0}^{m-1} p_i$ follows from (25). Thus, at $m=N$, the solution $(p_0,p_1,\cdots,p_m,\cdots,p_N)$ is obtained. Finally, from the balance equations, if $p_{m-2}\neq 0, p_{m-1}=0$, then $p_m=p_{m+1}=\cdots=p_N=0$. Thus equations (25) and (27) provide the recursive computational procedure for the calculation of the optimal control laws over the set of all admissible controls Λ.

VI. Concluding Remarks

The throughput time delay trade-off has been chosen as a criterion to find optimal flow control policies for a birth-death process. Two objective function that characterize this trade-off have been investigated. Both functions lead to mathematical programming problems. The optimal policies obtained are window flow control mechanisms. The results fulfill the practical expectations that the optimal flow control of a birth-death process is of a bang-bang type. They can be traced back to the structure of the detailed balance equations that the equilibrium probabilities satisfy.

Acknowledgement

The research reported here was supported in part by the National Science Foundation under Grant # NSF ECS-81-08776 and in part by the Army Research Office under Contract DAAG29-83-K-0021.

References

[1] Chandy, K.M., Herzog, U. and Woo, L., "Parametric Analysis of Queueing Networks", *IBM J. Research and Development*, Vol. **19**, No. 1, pp. 43-49, Jan. 1975.

[2] Kleinrock, L., Queueing Systems, Volume I: Theory, John Wiley, New York, 1975.

[3] Larson, R. E., and Casti, J. L., Principles of Dynamic Programming - Part I: Basic Analytic and Computational Methods, Marcel Dekker, Inc., New York, 1978.

[4] Lazar, A. A., "Optimal Control of M/M/1 Queue", *Proceeding of the 19th Annual Allerton Conference on Communication, Control and Computing*, University of Illinois, Urbana-Champain, Ill., October 8-10, 1981.

[5] Lazar, A. A., "Optimal Flow Control of a Class of Queueing Networks in Equilibrium", *IEEE Transactions on Automatic Control*, Vol. AC-28, No. 10, October 1983.

[6] Lazar, A.A., "Centralized Optimal Control of a Jacksonian Network", *Proceedings of the Sixteenth Conference on Information Sciences and Systems*, Princeton University, Princeton N.J., March 17-19, October 1982.

[7] Luenberger, D. G., Introduction to Linear and Nonlinear Programming, Addison Wesley, Reading, MA, 1973.

[8] Schwartz, M., Computer-Communication Network Design and Analysis, Prentice-Hall, Englewood Cliffs, N.J., 1977.

[9] Simonnard, M., Linear Programming, translated by W. S. Jewell, Prentice-Hall, Englewood Cliffs, N.J., 1966.

[10] Weiss, G., Personal Communication, April 1982.

Flow Control in ISDN Based on the Estimation of the Voice Traffic Load

Faramak Vakil and *Aurel A. Lazar*
Columbia University
Department of Electrical Engineering
Computer Communications Research Laboratory

Abstract

The design of the optimal data link flow control protocol for Integrated Services Digital Networks with partially observed voice traffic is investigated. The model employed consists of a source node that controls a queueing system with random server. The controller has only incomplete knowledge about the state of the server. A recursive optimal m.m.s.e. filter is designed to estimate the state of the Markov chain modeling the voice traffic flow. An adaptive window flow control mechanism with a window size chosen according to the estimated voice traffic load is implemented to control the flow of the data packets. The performance of the data flow control strategy is also discussed.

I. Introduction

In an integrated voice/data network the data packets are transmitted through a network with random link capacities [4]. The optimal data link flow control can be modeled as the optimal control of a queueing system with random departure rate. This rate is governed by the voice traffic statistics in the system. The birth rate is chosen from the convex set of admissible controls.

With regard to the knowledge of the controller of the state of the voice traffic in the system two different information structures are distinguished. These lead to control strategies with complete and with partial observations [4]. In the partial observation case only the state of the data buffer $Q=(Q_t)$, $0 \leq t \leq T$, is available at the controller. This information can be obtained by continuously monitoring the departure and arrival processes associated with the destination node.

In this paper an optimal filter is implemented to supply an estimate of the voice traffic load to the controller. Our approach is based upon the separation principle [2]. The precise mathematical formulation is sketched below.

II. Data Link Flow Control with Partial Observations

Let $A=(A_t,F_t)$, $0 \leq t \leq T$, and $D=(D_t,F_t)$, $0 \leq t \leq T$, be the arrival and departure processes associated with the receiving queueing system. We have

$$Q_t = Q_0 + A_t - D_t$$

for all t, $0 \le t \le T$, where Q_0 is the initial state of the data buffer. The state of the voice traffic on the link, $X=(X_t)$, $0 \le t \le T$, is modeled as a (M-m+1) state birth-death process with birth and death rates $\eta=(\eta_k)$, $0 \le t \le T$, and $\zeta=(\zeta_k)$, $0 \le t \le T$, respectively [4], [5]. The service rate of the destination queue at time t, μ_t, is given by:

$$\mu_t = b\,(m + X_t) \;.$$

The estimation of the voice traffic load is analytically equivalent to the estimation of the state of the Markov chain X_t driving the doubly stochastic Poisson process $D_t 1(Q_t > 0)$. The recursive equation for the computation of the conditional probability distribution of the Markov chain X_t is given by [1] (see also [3]):

$$Z(t) = Z(0) + \int_0^t G Z(u)\,du + \int_0^t [-Z(u^-) + \frac{HZ(u^-)}{FZ(u^-)}][dD_u - FZ(u)\,du] \quad ,$$

where G is the infinitesimal generator matrix of the Markov chain X_t and $Z(t)=[z_k(t)]$ is a (M-m+1)x1 vector with elements $z_k(t)=P[X_t=k \mid F_t]$. $H=[h_{ij}]$ is a (M-m+1)x(M-m+1) diagonal matrix with elements $h_{kk}=b\,(M-k)$ and $F=[f_k]$ is an (M-m+1)x1 vector with $f_k=b\,(M-k)$. Thus, the optimum m.m.s.e. estimate of the voice traffic load is given by :

$$\hat{X}_t = E[X_t \mid F_t] = \sum_{k=0}^{M-m} k z_k(t)$$

Intuitively, therefore, the optimal flow control with partial observation has been reduced to a flow control strategy with complete observations (with $\hat{X}_t$ substituting for X_t). As in [4] it only remains to implement the adaptive window flow control mechanism.

Acknowledgement

The research reported here was supported in part by the National Science Foundation under Grant # ECS-81-08776 and in part by the Office of Naval Research under Contract N00014-82-K-0508.

References

[1] Bremaud, P., Point Processes and Queues : Martingale Dynamics, Springer Verlag, New York,1981.

[2] Rishel, R., "The Minimum Principle, Separation Principle, and Dynamic Programming for Partially Observed Jump Processes", *IEEE Transactions on Automatic Control*, Vol. AC-23, Dec. 1978.

[3] Snyder, D.L., "Filtering and Detection for Doubly Stochastic Poisson Processes", *IEEE Transactions on Information Theory*, Vol IT-18, No. 1, Jan. 1972.

[4] Vakil, F. and Lazar, A.A., " Towards Modeling and Dynamic Optimal Flow Control of Integrated Services Digital Networks (I)", *Proceedings of the* 1983 *Conference on Information Science and Systems*, John Hopkins University, Baltimore, MD 21218, March 23-25, 1983.

[5] Weinstein, C.J., Malpass, M.L. and Fisher, M.J., "Data Performance of an Integrated Circuit and Packet Switched Multiplex Structure", *Proceedings of the International Conference on Communications*, 1979, pp. 24.3.1 - 24.3.5 .

AGGREGATION AND DECOMPOSITION IN THE PLANNING OF EVOLVING TELECOMMUNICATIONS NETWORKS

ALEXANDER D. KASAVIN and JOHN R. MYERS
GTE Sprint Communications, Burlingame, CA

Certain well-known models and design techniques have been used for the planning of the telecommunications networks in static cases, when the number of nodes in a network is relatively small and does not change much over time. However, those models lose their efficiency and eventually fail in a dynamic situation when the number of nodes and interconnect links as well as the volume of processed traffic information, grows rapidly. Currently, such a dynamic situation is typical for the U.S. long distance telecommunications industry.

Three practical system problems, related to the planning of the growing GTE Sprint telecommunications network will be considered. Some approaches to the solution of these problems, based on the matrix aggregation techniques (1) and the system decomposition principle (2), will be discussed.

1. Design of the switch network hierarchy.

   The idea of this design is to match the switch hierarchy with a natural structure of the geographical traffic distribution throughout the network. This structure is unknown in advance, but can be observed from a measured point-to-point (city-to-city) traffic matrix by virtue of a matrix aggregation algorithm.

   The algorithm clusters cities so that, pairs of cities within a cluster would, on the average, interact more than pairs of cities from different clusters.

   Such clusters, observed for different ranges of traffic levels, together with these ranges, define the structure of the traffic distribution.

2. Long-range traffic forecast.

   In mathematical terms, this is a problem of forecasting a point-to-point traffic matrix whose dimensions are expanding over time. With the number of cities in the network growing at some constant rate, the number of matrix elements increases as the square of the number of cities.

Analysis of the observed structure of the traffic distribution suggests a forecasting technique which utilizes a certain aggregation of historical point-to-point traffic data. This fixes the dimensions of the forecast matrices and increases the accuracy of the forecast.

3. Architectural design of a long-range planning support system.

There are two system levels of the network representation in the planning process: a switch network which determines a traffic routing strategy, and a transmission network which describes the paths and capacity of actual, physical transmission facilities. Traditionally, one and the same planning routine with one and the same 'fine-granulated' point-to-point traffic information as the input, has been used for both short-range and long-range planning.

However, the two subsystems evolve in two different time scales. The switch network evolution is subject to a fast dynamics whereas the changes in transmission network, occur at a much slower rate. This observation supports our design idea to decouple the planning of these two subsystems and to simplify the long-range planning routine by eliminating unnecessarily detailed information from this process.

Implementation of this idea requires re-evaluation of the architecture of the entire planning system and the design of appropriate new support models.

The results of preliminary testing and modeling of the proposed approaches prove their efficiency in the evolutionary planning of the GTE Sprint Network.

References

1. E.M.Braverman, A.A.Dorofeyuk, V.J.Lumelsky and I.B. Muchnik (1971) "Diagonalization of a Coupling Matrix and Detection of Hidden Factors." In: Proceeding of the Institute of Control Sciences, No. 1, pp 42-69

2. P.V.Kokotovic (1981) Subsystems, Time Scales and Multi-modeling. Automatica, Vol. 17, No. 6, pp 789-795

A Mixed-Integer Linear Programming Problem Which Is Efficiently Solvable

CHARLES E. LEISERSON
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

JAMES B. SAXE
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

Abstract—Efficient algorithms are known for the simple linear programming problem where each inequality is of the form $x_j - x_i \leq a_{ij}$. Furthermore, these techniques extend to the pure integer programming variant of the problem where all the unknowns are required to be integers. This paper gives an efficient solution to the mixed-integer linear programming variant where some, but not necessarily all, of the unknowns are required to be integers. The algorithm we develop is based on a graph representation of the constraint system and runs in $O(|V||E| \lg |V|)$ time. It has several applications including optimal retiming of synchronous circuitry, VLSI layout compaction in the presence of power and ground buses, and PERT scheduling with periodic constraints.

1. Introduction

Much research has centered on the problem of finding shortest paths in graphs. It is well known that there is a direct correspondence between the ***single-source shortest-paths problem*** and the following simple linear programming problem.

> *Let S be a set of linear inequalities of the form $x_j - x_i \leq a_{ij}$, where the x_i are unknowns and the a_{ij} are given real constants. Determine a set of values for the x_i such that the inequalities in S are satisfied, or determine that no such values exist.*

This paper considers the *mixed-integer* linear programming variant of this problem in which some (but not necessarily all) of the x_i are required to be integers. The problem arises in the context of synchronous circuit optimization [8], but it has applications to PERT scheduling and VLSI layout compaction as well.

Before formally defining the mixed-integer programming problem, we restate the linear programming problem above in another form.

> **Problem L.** *Let $G = (V, E, a)$ be an edge-weighted, directed graph, where $V = \{1, 2, \ldots, |V|\}$ is the vertex set, the set E of edges is a subset of $V \times V$, and for each edge $(i, j) \in E$ the edge weight a_{ij} is a real number. Find a vector $x = (x_1, x_2, \ldots, x_{|V|})$ satisfying the constraint that*
>
> $$x_j - x_i \leq a_{ij}$$
>
> *for all $(i, j) \in E$, or determine that no feasible vector exists.*

The graph G is called a *constraint graph* for the linear programming problem. There are three advantages in adopting a graph representation of the problem. First, an adjacency-list representation [1, p. 200] of the constraint graph G is more economical than, for example, a linear programming tableau or, when the graph has relatively few edges, a matrix of the a_{ij}. Second, Problem L frequently arises in situations that are naturally described by a graph. Finally, the graph-theoretic formulation helps in understanding the algorithms that solve this kind of problem.

A method for solving Problem L was discovered in the late 1950's by Ford and Bellman [7, p. 74]. Yen [12] gave some improvements to the Bellman-Ford algorithm as well as a cogent

---

This research was supported in part by the Defense Advanced Research Projects Agency under Contract N00014-80-C-0622 and by the Office of Naval Research under Contract N00014-76-C-0370.

analysis showing that its running time is $O(|V|^3)$. This bound is easily improved to $O(|V||E|)$ by using an adjacency-list representation for the constraint graph.

The Bellman-Ford algorithm can also be used to solve the *integer* linear programming variant of Problem L, in which all the x_i are required to be integers. If the edge weights a_{ij} all happen to be integers, the Bellman-Ford algorithm will produce integer values for the x_i. If the a_{ij} are not integers, however, but the x_i are required to be integers, each edge weight a_{ij} may be replaced by $\lfloor a_{ij} \rfloor$ without affecting the satisfiablity of the inequalities.

The focus of this paper is the *mixed-integer* variant of Problem L.

Problem MI. *Let $G = (V, V_I, E, a)$ be a "mixed-integer," edge-weighted, directed graph, where $V = 1, 2, \ldots, |V|$ is the vertex set, the set V_I is a subset of V, the set E of edges is a subset of $V \times V$, and for each edge $(i,j) \in E$ the edge weight a_{ij} is a real number. Find a vector $x = (x_1, x_2, \ldots, x_{|V|})$ satisfying the constraints that*

$$x_j - x_i \leq a_{ij}$$

for all $(i,j) \in E$ and that $x_i \in \mathbf{Z}$ for all $i \in V_I$, or determine that no feasible vector exists.

The vector $x = (x_1, x_2, \ldots, x_{|V|})$ is called a *solution* to graph G, and if graph G has a solution, we say that G is *satisfiable.* When it is clear from context, we use the same terminology for Problem L.

In addition, we shall refer to the vertices in V_I as the *integer* vertices of G and the vertices in $V_R = V - V_I$ as the *real* vertices of G. We also partition the set of edges into two sets depending on whether the vertex at the head of the edge is integer or real:

$$E_I = \{(i,j) \in E \mid j \in V_I\},$$
$$E_R = \{(i,j) \in E \mid j \in V_R\}.$$

This paper presents two algorithms to solve Problem MI. The first, which runs in $O(|V|^2|E|)$ time, is a straightforward extension of the Bellman-Ford algorithm. The second is more sophisticated and has a running time of $O(|V||E| \lg |V|)$ for arbitrary graphs and $O(|V||E|)$ for dense graphs. We conjecture that the $O(|V||E|)$ running time achieved by the Bellman-Ford algorithm for the pure linear programming and pure integer programming versions of the problem is not achievable in general for Problem MI.

The remainder of this paper is organized as follows. Section 2 reviews the Bellman-Ford algorithm. Section 3 presents a simple relaxation algorithm for solving Problem MI. Section 4 discusses two techniques—Dijkstra's algorithm and *reweighting*—which are used in Section 5 to construct an asymptotically efficient algorithm for Problem MI. We discuss applications and present some concluding remarks in Section 6.

2. Shortest paths and the Bellman-Ford algorithm

This section reviews how the *Bellman-Ford algorithm* solves Problem L. Although the results of this section are well known and can be found in most textbooks on combinatorial optimization (see, for example, [7, p. 74]), we repeat the material here for the reader's convenience.

There is a natural correspondence between Problem L and the graph-theoretic *single-source shortest-paths* problem. Let $G = (E, V, a)$ be an instance of Problem L. Suppose that for each vertex $i \in V$, there is a path to i from vertex 1, and let d_i be the weight of shortest (least-weight) path from vertex 1 to vertex i. (At the end of the section, we shall discuss the case in which some vertices are not reachable from vertex 1.) Then for any edge $(i,j) \in E$, we have $d_j - d_i \leq a_{ij}$ since the edge (i,j) can be appended to a shortest path from vertex 1 to vertex i to produce a path from vertex 1 to vertex j of weight $d_i + a_{ij}$. Thus the shortest-path weights d are a solution to G.

Whenever G is satisfiable, there are infinite number of solutions. For example, if x is a solution to G, then uniformly adding any constant k to each x_i yields another solution y, where

$y_i = x_i + k$ for each $i \in V$. The assignment $x_i \leftarrow d_i$ gives each x_i its largest possible value subject to the constraint that $x_1 = 0$. To see this, consider any path p of weight d_i from vertex 1 to vertex i. If the inequalities associated with the edges of p are summed, the unknowns associated with the intermediate vertices cancel and the result is the inequality $x_i - x_1 \leq d_i$.

Whenever the graph G contains some cycle c whose weight is negative, the shortest path weight from vertex 1 to any vertex i on cycle c is undefined because the weight of any path to vertex i can be diminished by appending a traversal of c. In this case the graph G is not satisfiable. If the inequalities associated with the edges of c are summed, all the unknowns x_i cancel, and the resulting inequality asserts that 0 is less than or equal to the weight of c, which is false.

The Bellman-Ford algorithm, which is given below, solves Problem L by finding the weight of the shortest path to each vertex from vertex 1. Should the graph contain a negative-weight cycle, the algorithm reports that the graph is unsatisfiable by calling the procedure *Fail,* whose semantics we leave unspecified.

Algorithm BF *(Bellman-Ford algorithm).*

```
BF1.  $x_1 \leftarrow 0$;
BF2.  for $i \leftarrow 2$ to $|V|$ do $x_i \leftarrow \infty$;
BF3.  for ind $\leftarrow 1$ to $|V| - 1$ do
BF4.      foreach $(i, j) \in E$ do
BF5.          if $x_j > x_i + a_{ij}$ then $x_j \leftarrow x_i + a_{ij}$;
BF6.  foreach $(i, j) \in E$ do
BF7.      if $x_j > x_i + a_{ij}$ then Fail
```

For each vertex $j \in V$, the Bellman-Ford algorithm iteratively updates the weight x_j of a tentative shortest path from vertex 1 to vertex j. After initialization, the algorithm makes $|V| - 1$ passes through the edges in E and *relaxes* each edge (i, j) by computing $x_j \leftarrow \min(x_j, x_i + a_{ij})$.

A simple analysis due to Yen [12] indicates why the Bellman-Ford algorithm works. The weight x_j converges to the weight d_j of a shortest path from vertex 1 to vertex j if the edges on the path are relaxed in order along the path. The sequence of edges relaxed by the Bellman-Ford algorithm consists of $|V| - 1$ copies of some ordering of E, and therefore contains every vertex-disjoint path as a subsequence. If there are no negative-weight cycles in G, then every shortest path is vertex disjoint, so each x_i converges to the shortest-path weight d_i. On the other hand, if there is a negative-weight cycle in the graph, the algorithm detects this condition by iterating once more through all edges to see whether any of the inequalities remain unsatisfied.

The Bellman-Ford algorithm as given above determines the weight of the shortest path from vertex 1 to each vertex, and therefore solves Problem L whenever all vertices of G are reachable from vertex 1. The code can be adapted to solve Problem L on arbitrary graphs by simply changing the initialization step (lines BF1–BF2). In particular, if each x_i is assigned a finite initial value u_i, the relaxation in lines BF3–BF5 sets each x_i to its maximum value subject to the constraints that $x_j - x_i \leq a_{ij}$ for each edge $(i, j) \in E$ and that $x_i \leq u_i$ for each vertex $i \in V$. Notice that whenever the constraint graph G is satisfiable, it is satisfiable subject to the additional constraints $x_i \leq u_i$. Should the inequalities be inconsistent because there is a negative-weight cycles in the graph, the relaxation will not converge to a solution, and the inconsistency will be detected by the test in lines BF6–BF7.

3. Simple relaxation algorithms for Problem MI

As was mentioned in the introduction, Problem MI can be solved directly by the Bellman-Ford algorithm when all unknowns are real (Problem L) and when all unknowns are integer. The combination of integer and real unknowns, however, seems to make the problem harder. In this section, we gain some intuition about the structure of Problem MI by introducing two

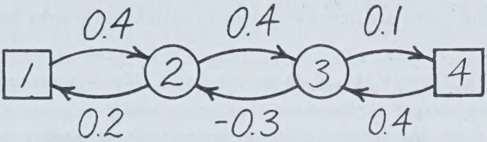


Figure 1. An instance of Problem MI. Integer vertices ($V_I = \{1, 4\}$) are shown as squares and the real vertices as circles.

algorithms that solve it in much the same way as the Bellman-Ford algorithm solves Problem L. The asymptotically efficient algorithm from Section 4 is derived from the second of these algorithms.

A natural approach to solving Problem MI is to see whether the Bellman-Ford relaxation approach can be made to work. Since we have both integer and real vertices in the graph, however, we must modify the relaxation step BF5 in the Bellman-Ford algorithm to produce an integer value whenever j is an integer vertex (line R6). This approach does in fact work, but it requires more iterations than the simple Bellman-Ford algorithm. The next algorithm solves Problem MI. The number of iterations n in line R2 will be determined in the analysis following the algorithm.

Algorithm R. *(Relaxation.)*

```
R1.  foreach $i \in V$ do $x_i \leftarrow 0$;
R2.  for $ind \leftarrow 1$ to $n$ do
R3.      foreach $(i, j) \in E$ do
R4.          begin
R5.          $x_j \leftarrow \min(x_j, x_i + a_{ij})$;
R6.          if $j \in V_I$ then $x_j \leftarrow \lfloor x_j \rfloor$;
R7.          end;
R8.  foreach $(i, j) \in E$ do
R9.      if $x_j > x_i + a_{ij}$ then $Fail$;
```

In order to determine a value of n such that Algorithm R works, we introduce the notion of a *reducing path.* Let p be a path starting at some vertex k, and suppose that x_k is initially set to 0 and that all the remaining x_i are initialized to ∞. Suppose the edges in path p are traversed in order starting from k, and each edge (i, j) along the path is relaxed as in statements R5–R6. If each relaxation of an edge (i, j) reduces the value x_j, the path p is called a ***reducing path.***

Whenever a sequence of edges contains all reducing paths as subsequences, the relaxation of each edge in the sequence in order yields a solution. (The proof is analogous to Yen's analysis [12] of the Bellman-Ford algorithm.) The Bellman-Ford algorithm solves Problem L because in a satisfiable graph with only real vertices, each vertex occurs at most once on any single reducing path. (And in fact, every shortest path is a reducing path.)

When some unknowns are integer and some are real, however, it is possible for a reducing path to visit the same vertex more than once, even if the graph is satisfiable. For example, in the graph shown in Figure 1, the reducing path $p = 3 \rightarrow 2 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 3 \rightarrow 2$ visits vertices 2 and 3 three times each. If all the x_i are initially set to 0, the edges of p must be relaxed in their order along the path to achieve convergence. Moreover, relaxing the entire edge set in some arbitrary order only $3 = |V| - 1$ times might not achieve convergence. Since the value of n in line R2 must be at least the maximum number of edges in any reducing path, the value $|V| - 1$, which was used in Algorithm BF, will not suffice.

Fortunately, reducing paths are never very long in satisfiable graphs because of the following lemma.

Lemma 1. *Suppose $G = (V, V_I, E, a)$ is satisfiable. If p is a reducing path in G, then*

1. *p visits no integer vertex more than once, and*
2. *p never visits the same real vertex twice without visiting some integer vertex in between.*

Proof. If either condition is violated, then the reducing path p can be extended indefinitely by repeating the cycle that causes violation. ▮

Lemma 1 allows us to determine a value for n in line R2 of Algorithm R such that the x converges to a solution whenever G is satisfiable. Any reducing path contains each integer vertex at most once and each real vertex at most $|V_I| + 1$ times. Since the number of edges in a path is one less than the number of vertices, any reducing path for a satisfiable graph can have no more than $|V_I| + (|V_I| + 1)|V_R| - 1 = |V_I||V_R| + |V| - 1$ edges. Thus the limit n of the outer loop in Algorithm R should be set to $|V_I||V_R| + |V| - 1$.

This analysis suggests the following algorithm which is slightly more efficient than Algorithm R, and which forms the basis of the asymptotically efficient algorithm presented in the next section.

Algorithm M. *(Modified relaxation.)*

```
M1.   foreach $i \in V$ do $x_i \leftarrow 0$;
M2.   for ind ← 1 to $|V_R|$ do
M3.       foreach $(i, j) \in E_R$ do
M4.           $x_j \leftarrow \min(x_j, x_i + a_{ij})$;
M5.   for ind2 ← 1 to $|V_I|$ do
M6.       begin
M7.       foreach $(i, j) \in E_I$ do
M8.           $x_j \leftarrow \min(x_j, \lfloor x_i + a_{ij} \rfloor)$;
M9.       for ind ← 1 to $|V_R|$ do
M10.          foreach $(i, j) \in E_R$ do
M11.              $x_j \leftarrow \min(x_j, x_i + a_{ij})$;
M12.      end;
M13.  foreach $(i, j) \in E$ do
M14.      if $x_j > x_i + a_{ij}$ then Fail;
```

The only difference between this algorithm and Algorithm R is that it treats the edges in E_I separately from the edges in E_R. In lines M7–M8 of Algorithm M, each edge in E_I is relaxed once. There are $|V_I|$ such passes over E_I which are preceded, followed, and separated by *exhaustive relaxations* of the edges in E_R (lines M2–M4 and M9–M11). In each exhaustive relaxation of E_R, edges are relaxed until no further changes in the values of x_j are possible for $j \in V_R$. (Actually, the relaxations in lines M2–M4 and M9–M11 are only guaranteed to be exhaustive if there are no negative-weight cycles in E_R. If there are cycles of negative weight, however, this condition is detected at the end by the convergence test in lines M13–M14.)

4. Dijkstra's algorithm and reweighting

Section 5 gives a more efficient algorithm to solve Problem MI than either Algorithm R or Algorithm M. Two important techniques are used in the algorithm. The first is Dijkstra's algorithm which finds shortest paths in a graph from a single source in the case when all the edge weights are nonnegative. The other is *reweighting,* which is a technique due to Edmonds and Karp [3] and used by Johnson [6] in his efficient algorithm for solving the *all-pairs shortest-paths problem.*

Given a graph $G = (V, E, a)$ such that all edge weights a_{ij} are nonnegative, Dijkstra's algorithm computes for each vertex i, the weight d_i of the shortest path from vertex 1. Because each edge is relaxed exactly once, this algorithm is faster than the Bellman-Ford algorithm which solves the same problem for arbitrary edge weights. Dijkstra's algorithm derives its efficiency from the observation that along any shortest path from vertex 1, the shortest-path weights d_i form a nondecreasing sequence if all the edge weights are nonnegative. Thus, a sequence consisting of all edges $(i, j) \in E$ in nondecreasing order of the distances d_i contains as subsequences shortest paths from vertex 1 to all vertices in V. Furthermore, such a sequence of edges can be computed on the fly using a priority queue. (The textbook [1] gives a proof of correctness for this algorithm.)

Algorithm D *(Dijkstra's algorithm).*

```
D1.   $x_1 \leftarrow 0$;
D2.   for $i \leftarrow 2$ to $|V|$ do $x_i \leftarrow \infty$;
D3.   $Q \leftarrow V$;
D4.   while $Q \neq \emptyset$ do
D5.       begin
D6.       Choose $i \in Q$ such that $x_i = \min_{j \in Q} x_j$;
D7.       $Q \leftarrow Q - \{i\}$;
D8.       foreach $j \in V_R$ such that $(i, j) \in E_R$ do
D9.           $x_j \leftarrow \min(x_j, x_i + a_{ij})$;
D10.      end;
```

If the set Q in the algorithm is implemented as a standard priority queue, each extraction (lines D5–D6) and update (line D8) costs $O(\lg |Q|) = O(\lg |V|)$ time. Thus the total running time of Dijkstra's algorithm is $O(|E| \lg |V|)$. Johnson [6] shows that by implementing Q as a fixed-height heap [5], the running time can be brought down to $O(h|E| + h|V|^{1+1/h})$, where h is an integer constant that may be chosen after the input is presented. The choice $h = \lceil \lg |V| \rceil$ gives the bound $O(|E| \lg |V|)$. For families of *dense* graphs where $|E| = \Omega(|V|^{1+\epsilon})$ for some constant $\epsilon > 0$, the choice $h = \lceil 1/\epsilon \rceil$ gives an $O(|E|)$ bound.

Since Dijkstra's algorithm is equivalent to the Bellman-Ford algorithm on graphs with nonnegative edge weights, it can be used to solve Problem L on such graphs. This is not very interesting in itself, since any graph $G = (V, E, a)$ in which all edge weights are nonnegative can be trivially satisfied by setting x_i to 0 for each $i \in V$. Our interest in Dijkstra's algorithm comes from a stronger property of the solutions it finds. Suppose the initialization step (lines D1–D2) is changed so that each variable x_i is initialized to a finite value u_i. Then the relaxation procedure in lines D3–D10 will set each x_i to its largest possible value consistent with the constraints that $x_j - x_i \leq a_{ij}$ for each edge $(i, j) \in E$ and that $x_i \leq u_i$ for each vertex $i \in V$. In other words, lines D3-D10 of Dijkstra's algorithm are functionally equivalent to lines BF3-BF5 of the Bellman-Ford algorithm provided that all the edge weights a_{ij} are nonnegative. Since a graph with only nonnegative edge weights can never contain a negative-weight cycle, no test for convergence is necessary in this case.

The efficient algorithm we shall present to solve Problem MI is a modification of Algorithm M. Notice that lines M9–M11 of Algorithm M exhaustively relax the edges in E_R in a manner similar to lines BF2–BF4 of the Bellman-Ford algorithm. In Algorithm M, however, this code is executed many times. The efficient algorithm to solve Problem MI uses a trick to replace this code with code based on the more efficient relaxation procedure in lines D3-D10 of Dijkstra's algorithm. This trick is the technique of *reweighting* due to Edmonds and Karp [3].

Lemma 2. *Let $G = (V, E, a)$ be an edge-weighted graph, for each $i \in V$ let r_i be a real number, and let $H = (V, E, b)$ where $b_{ij} = a_{ij} + r_i - r_j$ for each edge $(i, j) \in E$. For each vertex $i \in V$ let x_i be a real number and let $y_i = x_i - r_i$. Then $x_j - x_i \leq a_{ij}$ for all $(i, j) \in E$ if and only if $y_j - y_i \leq b_{ij}$ for all $(i, j) \in E$ (that is, x is a solution to G if and only if y is a solution to H.)*

Proof. Trivial. ∎

We call the vector $r = (r_1, r_2, \ldots, r_{|V|})$ a *reweighting* of the graph G.

5. An asymptotically efficient algorithm for solving Problem MI

This section shows how Dijkstra's algorithm and reweighting can be incorporated into Algorithm M to yield a faster algorithm for solving Problem MI. Given a graph $G = (V, V_I, E, a)$, the idea is to find a reweighting r such that the reweighted graph $H = (V, V_I, E, b)$ has edge weights $b_{ij} = a_{ij} + r_i - r_j \geq 0$ for all edges $(i,j) \in E_R$. Lemma 2 guarantees that G is satisfiable if and only if H is satisfiable and also that a solution y to H can be converted into a solution x to G by setting $x_i = y_i + r_i$ for each $i \in V$. The advantage gained by transforming the problem on G to a problem on H is that the relaxation portion of Dijkstra's algorithm (lines D3–D10) can replace the Bellman-Ford relaxation (lines M9–M11), which is the most expensive part of Algorithm M.

The first stage of the algorithm is to determine the reweighting values r_i for all $i \in V$ and the new edge weights $b_{ij} = a_i + r_i - r_j$ for all $(i,j) \in E$. We must choose the values r_i such that $b_i \geq 0$ for all $(i,j) \in E_R$. Since this is equivalent to requiring that $r_j - r_i \leq a_{ij}$ for all $(i,j) \in E_R$, values for the r_i can be found by applying the Bellman-Ford algorithm to the graph (V, E_R, a). The first few lines of the algorithm are:

Algorithm T. *(Efficient algorithm.)*

```
T1.   for $i \in V$ do $r_i \leftarrow 0$;
T2.   for $ind \leftarrow 1$ to $|V_R|$ do
T3.       for $(i,j) \in E_R$ do
T4.           $r_j \leftarrow \min(r_j, r_i + a_{ij})$;
T5.   for $(i,j) \in E_R$ do
T6.       if $r_j > r_i + a_{ij}$ then Fail
T7.   for $(i,j) \in E$ do
T8.       $b_{ij} \leftarrow a_{ij} + r_i - r_j$;
```

If the algorithm fails in line T6, then there is a cycle of negative weight among the edges in E_R, and hence graph G is unsatisfiable even in the absense of integer constraints. Otherwise, the values b_{ij} computed in line T8 are nonnegative for all $(i,j) \in E_R$.

The next stage of Algorithm T is to solve the mixed-integer problem on the graph $H = (V, V_I, E, b)$ by alternately relaxing the edges in E_I and the edges in E_R as in Algorithm M. We begin by initializing the values y_i, which will converge to a solution to H if H is satisfiable.

```
T9.   for $i \in V$ do $y_i \leftarrow 0$;
```

This initialization has the added fortune of subsuming the first exhaustive relaxation of E_R (lines M2–M4 in Algorithm M). After the execution of line T9 we have $y_j - y_i = 0 - 0 \leq b_{ij}$ for all $(i,j) \in E_R$, which means that the edges in E_R are already exhaustively relaxed.

The next portion of Algorithm T parallels lines M5–M11 of Algorithm M and is where most of the computing gets done.

```
T10.  for $ind \leftarrow 1$ to $|V_I|$ do
T11.      begin
T12.      for $(i,j) \in E_I$ do
T13.          $y_j \leftarrow \min(y_j, \lfloor y_i + b_{ij} \rfloor)$;
T14.      $Q \leftarrow V$;
```

```
T15.    while $Q \neq \emptyset$ do
T16.        begin
T17.        Choose $i \in Q$ such that $y_i = \min_{j \in Q} y_j$;
T18.        $Q \leftarrow Q - \{i\}$;
T19.        for $j \in V_R$ such that $(i,j) \in E_R$ do
T20.            $y_j \leftarrow \min(y_j, y_i + b_{ij})$;
T21.        end;
T22.    end;
```

This code solves the problem on graph H in almost exactly the same way that Algorithm M would. The only difference is the method by which the edges of E_R are exhaustively relaxed. Whereas lines M9–M11 of Algorithm M perform the exhaustive relaxation using the Bellman-Ford algorithm, lines T14–T21 of Algorithm T take advantage of the nonnegativity of the b_{ij} for $(i,j) \in E_R$ and use Dijkstra's algorithm.

The final part of Algorithm T is to check the convergence of the y and to apply Lemma 2 to produce a satisfying assignment x for the original graph G.

```
T23.  for $(i,j) \in E_I$ do
T24.      if $y_j > y_i + b_{ij}$ then Fail;
T25.  for $(i,j) \in E$ do
T26.      $x_i \leftarrow y_i + r_i$;
```

Lines T23–T24 check the convergence of y by testing the inequalities associated with the edges in E_I. The inequalities resulting from edges in E_R need not be checked because the relaxation in lines T14–T22 is guaranteed to be exhaustive. (If there were negative-weight cycles in E_R, we would have detected this in lines T5–T6.)

Lines T25–T26 convert the solution y to graph H into a solution x to graph G. Lemma 2 ensures that the inequalities $x_j - x_i \leq a_{ij}$ are satisfied, but we must also show that the x_i are integers for all $i \in V_I$. For each $i \in V_I$ the value y_i is an integer, however, and furthermore, the values of the r_i produced in lines T1–T4 are zero for all $i \in V_I$. Thus for all the integer vertices, the x_i are integers.

In summary, we have proved the following theorem.

Theorem 3. *Algorithm T solves Problem MI.*

The running time of Algorithm T is $O(|V||E| \lg |V|)$. (Johnson's techniques [6] [5] can be used to reduce the actual running time to $O(|V||E|)$ for dense graphs by implementing the priority queue Q as a fixed-height heap.) Tighter analysis in terms of the sizes of the sets V_I, V_R, E_I, and E_R is possible, however. In particular, the closer bound $O(|V_R||E_R| + |V_I||E_I| + |V_I||E_R| \lg |V|)$ indicates that the algorithm performs even better when the number of integer vertices is small.

6. Applications, extensions, and conclusions

The solution to Problem MI was demanded by a problem concerning optimization of sychronous circuitry by retiming [8]. This section briefly describes two other problems—compaction of VLSI circuits in the presence of power and ground buses and PERT scheduling with periodic constraints—which can be reduced to Problem L. We also consider an extension of Problem MI where multiple classes of periodic constraints must be satisfied. (For example, some of the x_i are required to be integers, and others to be exact multiples of an integer constant c.)

Circuit compaction

Optimal (one-dimensional) compaction of VLSI circuit layouts [4] is another application of the Bellman-Ford algorithm. Each layout feature is given a variable representing an x-coordinate, and the design rules are enforced using constraints of the form $x_j - x_i \leq a_{ij}$. It may be desirable, however, to allow feature i to be to the left of feature j or vice versa, but not to allow them to occupy the same position. Unfortunately, if one wishes to allow this kind of transposition of layout features, either optimality or performance must be sacrificed because the problem becomes NP-complete [9]. But for certain compaction problems arising in practice, transposition of layout features can be allowed.

Some design methodologies enforce the placement of power, ground, and clock to be at regular intervals. For example, one signal processing system [10] requires that these wires be repeated every 200λ, and that the width of all cells in the system be integer multiples of this distance. The designer is then constrained to build a new cell so that the layout features are tightly packed among the global wires. In this context, where some layout features may go on one side or the other of some global wire but may not overlap, the compaction problem can be formulated as Problem MI.

PERT scheduling

Suppose we have a constraint graph with vertices representing milestones in a project, and edge-weights indicating the timing constraints between milestones. Generally, the Bellman-Ford algorithm can be used to provide an optimal scheduling of the milestones. If a work day is from 9:00 a.m. to 5:00 p.m., however, we may not wish to schedule a one-hour job to start at 4:30 p.m. Advancing the job to the next day may cause an earlier job to be advanced as well if the two jobs are constrained to fall near each other. The problem of PERT scheduling with periodic constraints can be cast as Problem MI.

Intuitively, the mixed-integer formulation allows one to include for each job (1) a real variable representing the starting time of the job, and (2) an integer variable representing, say, noon on the day the job occurs. Thus one can include constraints which say, for example, "This job must start before 4:00 p.m. on the day it occurs."

Multiple periodic constraints

Suppose that in the PERT scheduling application mentioned above, we also wish to take into consideration constraints involving weekends. To do this, we would associate with each job a third variable representing, say, Sunday noon of the week during which the job occurs. We are then required to solve a variant of MIin which there are two classes of periodic constraints—some variables are required to be exact integers and others to be exact multiples of 7 while the remainder may have arbitrary real values.

The solution to this problem is based on the following algorithm for solving Problem MI. (We assume without loss of generality that $G = (V, V_I, E, a)$ is strongly connected).

Algorithm U

```
U1.  if $(V, E, a)$ contains a negative-weight cycle then Fail
     else foreach $(i, j) \in V_I \times V_I$ do
         $b_{ij} \leftarrow \lfloor$the least path weight from $i$ to $j$ in $(V, E, a)\rfloor$;
U2.  if $(V_I, V_I \times V_I, b)$ contains a negative-weight cycle then Fail
     else find an integer assignment $x$ on $V_I$ such that $x_j - x_i \leq b_{ij}$ for all $i, j \in V_I$;
U3.  Apply the Bellman-Ford algorithm to $(V, E_R, a)$ using the $x_i$ found in Step U2 as
     initial values for the integer vertices and infinite initial values for the real vertices;
```

Step U1 produces a graph $H = (V_I, V_I \times V_I, b)$ which is feasible if and only if G is feasible, Step U2 solves H if H is feasible, and Step U3 extends the solution from the set V_I of integer vertices to the entire vertex set V. Step U1 can be performed in $O(|V|^3)$ time by the Floyd-Warshall algorithm [7] or in $O(|V||E| + |V_I||E| \lg |V|)$ time by Johnson's algorithm [6]. Step U2 can be performed by the Bellman-Ford algorithm and takes time $O(|V_I|^3)$ because H is a complete graph. These costs dominate the cost of Step U3, which takes only $O(|V||E|)$ time.

Algorithm U extends naturally to the case in which there are multiple classes of periodic constraints, provided that each period (*e.g.*, 1 week) is an exact multiple of the next smaller period (*e.g.*, 1 day). First, Step U1 is applied (with an appropriate scaling of the edge weights) to produce an equivalent problem in which the most loosely constrained class of vertices in the original problem is eliminated from consideration. This new problem is then solved recursively (or by direct application of Algorithm T if only two classes of vertices remain). Finally, the solution is extended to the entire set of vertices, as in Step U3.

Acknowledgments

We would like to acknowledge the contributions by Flavio Rose of MIT when we first studied this problem. The three of us originally produced Algorithm U, which is more thoroughly described in Flavio's master's thesis [11]. Thanks to Ron Rivest of MIT for reading an earlier draft of the paper. Thanks also to Don Johnson of Penn State, Dick Karp of Berkeley, Gene Lawler of Berkeley, and Nimrod Megiddo of CMU for helpful discussions.

References

[1] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman, *Data Structures and Algorithms,* Addison-Wesley, Reading, Massahusetts, 1983.

[2] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik,* Vol. 1, 1959, pp. 269–271.

[3] Jack Edmonds and Richard M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *Journal of the Association for Computing Machinery,* Vol. 19, No. 2, April 1972, pp. 248–264.

[4] Min-Yu Hsueh, "Symbolic layout and compaction of integrated circuits," Memorandum No. UCB/ERL M79/80, University of California, Berkeley, December 1979.

[5] Donald B. Johnson, "Priority queues with update and finding minimum spanning trees," *Information Processing Letters,* Vol. 4, No. 3, December 1975, pp. 53–57.

[6] Donald B. Johnson, "Efficient algorithms for shortest paths in sparse networks," *Journal of the Association for Computing Machinery,* Vol. 24, No. 1, pp. 1–13, January 1977.

[7] Eugene L. Lawler, *Combinatorial Optimization: Networks and Matroids,* Holt, Rinehart and Winston, New York, 1976.

[8] Charles E. Leiserson, Flavio M. Rose, and James B. Saxe, "Optimizing synchronous circuitry by retiming," *Third Caltech Conference on Very Large Scale Integration,* Randal Bryant, ed., Computer Science Press, Rockville, Maryland, March 1983, pp. 87–116.

[9] Thomas Lengauer, "On the solution of inequality systems relevant to IC-layout," *Proceedings of the 8th Conference on Graphtheoretic Concepts in Computer Science,* Carl Hänser Verlag, Munich, West Germany, 1982.

[10] Richard F. Lyon, "A bit-serial VLSI architectural methodology for signal processing," *VLSI '81,* John P. Gray, ed., Academic Press, New York, 1981, pp. 131–140.

[11] Flavio M. Rose, *Models for VLSI Circuits,* Masters Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, March 1982. Also available as MIT VLSI Memo No. 82–114.

[12] Jin Y. Yen, "An algorithm for finding shortest routes from all source nodes to a given destination in general networks," *Quarterly of Applied Mathematics,* Vol. 27, No. 4, 1970, pp. 526–530.

EDGE GRAMMARS AND PARALLEL COMPUTATION

FRANCINE BERMAN*
Purdue University

ABSTRACT

In this paper we introduce *edge grammars* as a tool for representing and embedding graph families. We discuss edge grammars in the formal language theory context and give conditions under which some graph families defined by edge grammars are self-embedding or *contractable*. We provide an algorithm for efficiently embedding large-sized graphs from some contractable graph families into smaller graphs in the family. A significant application of these results is to the problem of implementing large-sized problem instances of a parallel algorithm on a fixed-size parallel computer.

INTRODUCTION

Graph families are a fundamental tool in the theory of computation. They provide a visual representation of data structures (2-3 trees, heaps, shuffle-exchange graphs, etc. [AHU], [S3]), both sequential and parallel computation models [V], [P1], data flow of algorithms [RS], etc.

We usually think of a graph family as a set of the same kind of graph in various sizes, such as the family of complete binary trees or the family of cube-connected cycles. There have been a number of attempts to formalize this intuition by providing a generating device or a generating grammar for graph families analogous to the way languages are generated in automata theory. These generating formalisms include Lindenmayer systems [S1], graph grammars [CER], pair grammars [P2] and others. However the classes of graph families defined by such systems exclude many natural and useful graph families such as the family of shuffle-exchange graphs. In practice, the existing formalisms are not sufficient to characterize many of the graph families with the widest application.

In this paper, we define *edge grammars* as a tool for generating graph families and show how they can be used to define a wide class of graph families which include complete binary trees, square meshes, shuffle-exchange graphs, cube-connected cycles and other graphs used in data structures, algorithms and parallel computation. The application of edge grammars to parallel computation is especially lucrative. Graphs are a natural abstraction for the interconnection structure of a problem instance of a parallel algorithm and graph families are the proper abstraction for the interconnection structures of the set of all problem instances of a parallel algorithm. Representation of the graph family of a parallel algorithm by an edge grammar can be used to provide a uniform and efficient procedure for mapping large-sized problem instances of the algorithm into the interconnection structure of a fixed-size parallel computer. The mapping can then be used to multiplex the parallel algorithm on the fixed-size architecture in a straightforward way.

We discuss this application and give as an example the mapping of a parallel algorithm with a tree interconnection structure into a small parallel machine connected as a cube-connected cycle in Section 3. In Section 1, we define edge grammars and compare them with conventional constructs in automata theory. In Section 2, we discuss graph families which are self-embedding or *contractable* and show how the contraction and particularly the *truncation* of a graph family

---

*The work described herein is part of the Blue CHiP Project. It was funded in part by a Purdue Research Foundation Summer XL Grant.

can be used to provide a uniform mapping from larger to smaller graphs in the family. These results will prove especially useful to the parallel computation application in Section 3.

SECTION 1: EDGE GRAMMARS

Consider a graph family of the same kind of graph indexed by size. Assume that the same procedure builds each of the graphs in the family, varying only in the size of the parameters or the number of iterations of certain steps. We can capture the regularity of such a procedure by generating the graph family using *edge grammars*. In this section, we define edge grammars, compare and contrast them with conventional formal language constructs, and illustrate how they can be used to generate the families of complete binary trees, cube-connected cycles and shuffle-exchange graphs.

Definition: A *(Type 0) Edge Grammar* is a 4-tuple $\Gamma=<N,T,G,P>$ where T is a set of ordered pairs of strings over a finite alphabet Σ, N is a set of nonterminals, G in N is the start symbol, and $P \subseteq ((N \cup T)^* N (N \cup T)^*) X (N \cup T)^*$ is a finite set of productions. Define the concatenation of two edge sets A and B to be the set $AB=\{(vv',ww') | (v,w) \varepsilon A \text{ and } (v',w') \varepsilon B\}$.

Definition: An edge grammar is *Type 3* if the productions have the form $A \rightarrow BC$, $A \rightarrow B$ or $A \rightarrow C$ where A and B are nonterminals and C is a terminal; *Type 2* if the productions have the form $A \rightarrow BC$ or $A \rightarrow B$ where A is a nonterminal and B and C are either nonterminals or terminals; and *Type 1* if the number of terminals and nonterminals on the left-hand side of each production does not exceed the number of terminals and nonterminals on the right-hand side.

Definition: Let Γ be an edge grammar. The *nth graph generated by* Γ, $G(n)$, is the graph with vertices and edges

$$V(n)=\{v| \text{ For some } w, ((G \rightarrow {}^*(w,v) \text{ or } G \rightarrow {}^*(v,w)) \text{ and } |v|=|w|=n)\}$$
$$E(n)=\{(v,w)| \ G \rightarrow {}^*(v,w), \ v \neq w \text{ and } |v|=|w|=n\}.$$

Definition: Let Γ be an edge grammar. The ***graph family generated by*** Γ $(G(\Gamma))$ is the set $\{G(n)\}_{n>0}$ where $G(n)$ is the ***nth*** graph generated by Γ.

The usual language theoretic definitions such as *derivation* carry over in the edge grammar context with the constraint that a terminal pair (v,w) derived from the start nonterminal G is not in $G(\Gamma)$ unless the first and second coordinates are of the same length. The definitions of *Type 3*, *Type 2*, *Type 1* and *Type 0* are meant to draw the analogy between edge grammars and regular, context-free, context-sensitive and *Type 0* languages in the usual automata-theoretic sense [LP]. Much of the power of edge grammars, however, lies in the fact that they do not necessarily generate the products of *Type 0*, *1*, *2* and *3* languages. It is precisely the constraint that both coordinate vertices must be the same length which allows graph families such as the family of shuffle-exchange graphs which cannot be characterized within other graph formalisms to be defined with edge grammars. The connection between *Type 0*, *1*, *2* and *3* edge grammars and *Type 0*, *1*, *2* and *3* languages can be seen more precisely in Theorem 1.

Theorem 1

1) *For each $0 \leq x \leq 3$, any Type x language can be generated as the vertex set of a Type x edge grammar.*

2) *For $x=2,3$, there is a Type x edge grammar whose vertex set is not a Type x language.*

3) *For $x=0,1$, the vertex set of any Type x edge grammar is a Type x language.*

Proof 1)

Let L be a *Type x* language, $0 \leq x \leq 3$. Then there is a *Type x* grammar Γ which generates L. We construct an edge grammar Γ_E whose vertex set is L. For each terminal v which appears in a production of Γ, let (v,v) be a terminal in T_E. For each nonterminal A in Γ, let A be a nonterminal in N_E. The productions of Γ_E will be the productions of Γ with the pair (v,v) substituted for each terminal v. Clearly, the vertex set of the edge grammar Γ_E is the language L.

Proof 2)

Let Γ be the *Type 3* edge grammar with productions

$$\begin{array}{ll} G \to A(a,aa) & A \to A(a,aa) \\ A \to B(bb,b) & B \to B(bb,b) \\ B \to (\lambda,\lambda) & \end{array}$$

Then Γ generates pairs of strings of the form $(a^n b^{2m}, a^{2n} b^m)$, hence the vertex set of Γ is $V(n)=\{a^n b^{2n}\} \cup \{a^{2n} b^n\}$. By the pumping lemma [LP], $V(n)$ is not a *Type 3* language.

Similarly, let Γ' be the *Type 2* edge grammar with productions

$$\begin{array}{ll} G \to EF & \\ E \to AX & F \to YC \\ X \to EB_L & Y \to B_R F \\ E \to (\lambda,\lambda) & F \to (\lambda,\lambda) \\ B_L \to (b,\lambda) & B_R \to (\lambda,b) \\ A \to (a,a) & C \to (c,c) \end{array}$$

Then Γ generates pairs of strings of the form $(a^m b^m c^k, a^m b^k c^k)$. Such a string is only an edge in the graph family $\{G(n)\}$ when $m=k$. Again by the pumping lemma (for context-free languages), the vertex set $V(n)=\{a^m b^m c^m\}$ is not *Type 2*.

Proof 3)

To show that a *Type 0* language is the vertex set of a *Type 0* edge grammar, construct a Turing Machine with 3 tracks: one for the input, and one for each coordinate of the edge pairs generated from the *Type 0* edge grammar. Nondeterministically simulate the derivation of an edge pair on the second and third tracks. If the strings on tracks 2 and 3 consist only of terminal symbols and their lengths are the same, compare each of them with the input string. Accept the input (on the first track) if it matches the string on the second or the third track.

To show that the vertex set of a *Type 1* edge grammar Γ is a *Type 1* language, we again construct a 3 track Turing Machine which will nondeterministically simulate derivations within the edge grammar. Simulate only those pairs which can be derived within the number of cells used by the input string. If for some configuration, the strings on tracks 2 and 3 contain only terminal symbols and have the same length, compare each of them with the input string on track 1, accepting if one of them matches. By construction, the TM is linear-bounded and accepts precisely the set of strings in the vertex set of Γ.

Theorem 1 shows that *Type 2* and *3* graph families constitute a wider class than those graph families which can be generated by two-headed versions of the appropriate automata. The graph families generated by *Type 2* and *3* edge grammars alone inlcude many of the most frequently used graphs in algorithms, data structures, and parallel computation. For example, the families of square meshes, shuffle-exchange graphs [S3], toruses (meshes with wrap-around connections), binary n-cubes [NS], loops, 2-3 trees [AHU], cube-connected cycles [P1] and complete graphs are all generated by *Type 2* or *3* edge grammars. The following three examples give specific *Type 3* edge grammars which generate the families of cube-connected cycles and complete binary trees, and a *Type 2* edge grammar which generates the family of shuffle-exchange graphs.

Cube-Connected Cycles

Let $\Gamma=\langle\{G,H,L,C\},\{(1,1),(2,2),(3,3),(4,4),(1,3),(2,4),(3,4),(1,2)\},G,P\rangle$ where the productions are

$G\rightarrow H$
$H\rightarrow H(1,1)$
$H\rightarrow H(2,2)$
$H\rightarrow L(1,3)$
$H\rightarrow L(2,4)$
$H\rightarrow (1,2)$

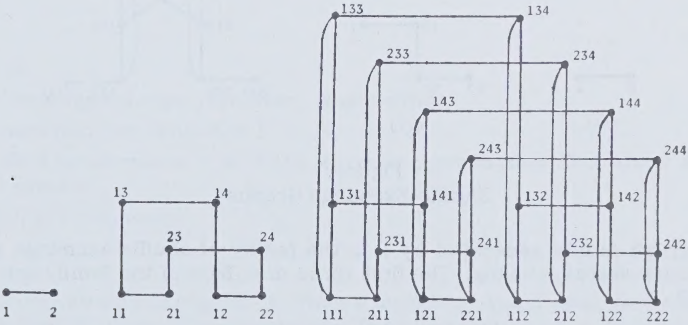
$H\rightarrow L(3,4)$
$C\rightarrow (2,2)$
$G\rightarrow C$
$C\rightarrow C(1,3)$
$C\rightarrow C(2,4)$
$C\rightarrow (1,1)$

$L\rightarrow (2,2)$
$L\rightarrow (1,1)$
$G\rightarrow L$
$L\rightarrow L(3,3)$
$L\rightarrow L(4,4)$

By applying the productions to the initial terminal pairs $H\rightarrow(1,2)$, $C\rightarrow(1,1)$, $C\rightarrow(2,2)$, $L\rightarrow(1,1)$, $L\rightarrow(2,2)$ given in P, we get

$H\rightarrow^{*}\{(1,2),(11,21),(12,22),(13,11),(21,23),(12,14),(22,24),(13,14),(23,24),\ldots\}$
$L\rightarrow^{*}\{(1,1),(2,2),(13,13),(23,23),(14,14),(24,24),\ldots\}$
$C\rightarrow^{*}\{(1,1),(2,2),(11,13),(21,23),(12,14),(22,24),\ldots\}$
$G\rightarrow H$, $G\rightarrow L$, $G\rightarrow C$.

The remaining pairs derived from G,H,L and C have coordinates with length 3 or greater resulting from longer derivations in Γ. The graph family generated by Γ is by definition the set of pairs generated from the nonterminal G with equal-length first and second coordinates. In particular, G(1), G(2) and G(3) are given in Figure 1.



G(1) G(2) G(3)

Figure 1
Cube-Connected Cycles

Complete Binary Trees

Let $\Gamma=<\{T_0,T_1,T_R,R,T\},\{(0,0),(1,1),(2,2),(2,0),(2,1)\},T,P>$ where the productions are

$$\begin{array}{lll}
T\to T_0 & T\to T_1 & T\to T_R \\
T_0\to T(0,0) & T_1\to T(1,1) & T_R\to R(2,0) \\
T\to(2,0) & T\to(2,1) & T_R\to R(2,1) \\
T\to R & R\to(2,2) & R\to R(2,2)
\end{array}$$

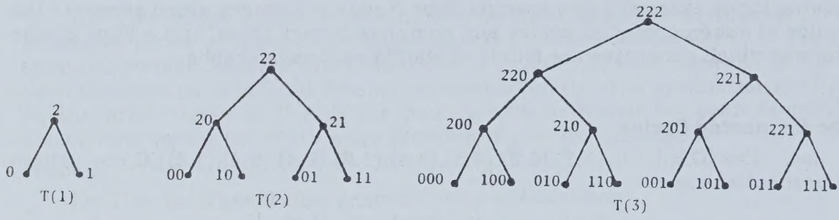

2
0
1
T(1)
22
20
21
00
10
01
11
T(2)
222
220
221
200
210
201
221
000
100
010
110
001
101
011
111
T(3)

Figure 2
Complete Binary Trees

The graph family generated by Γ is the family of complete binary trees. The first three graphs in the family are shown in Figure 2.

Shuffle-Exchange Graphs

Let $\Gamma=<\{G,N,A_0,A_1\},\{(0,0),(1,1),(0,1),(\lambda,0),(\lambda,1),(1,\lambda),(0,\lambda)\},G,P>$ where the productions are

$$\begin{array}{lll}
G\to A_0(\lambda,0) & A_0\to(0,\lambda) & N\to N(0,0) \\
G\to A_1(\lambda,1) & A_0\to A_0(0,0) & N\to N(1,1) \\
G\to N(0,1) & A_0\to A_0(1,1) & N\to(0,0) \\
G\to A_0 & A_1\to(1,\lambda) & N\to(1,1) \\
G\to A_1 & A_1\to A_1(1,1) & \\
G\to N & A_1\to A_1(0,0) &
\end{array}$$

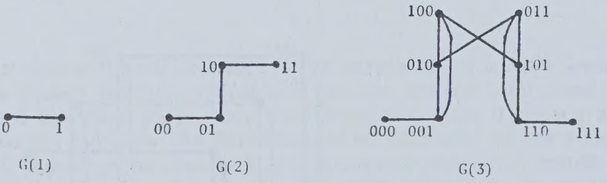

0
1
G(1)
10
11
00
01
G(2)
100
011
010
101
000
001
110
111
G(3)

Figure 3
Shuffle-Exchange Graphs

The graph family generated by Γ is the family of shuffle-exchange graphs with the usual vertex labeling. The first three members of the family are shown in Figure 3.

SECTION 2: CONTRACTABLE GRAPH FAMILIES

In many applications, it is desirable to be able to embed large graphs from a graph family into small graphs from the same family. When such an embedding can be done in a uniform fashion over the whole graph family, we call the family *contractable*. In this section, we focus on the subclass of contractable graph families for which the labels of the vertices can be used to provide a uniform contraction over the graph family. We call these graph families *truncatable*.

The notion of truncatable graph families will prove to be quite useful when the graph families are interpreted as interconnection graphs indexed by problem size for a given parallel algorithm. In the next section, we describe this application. In this section, we define contractability and truncatability and give conditions on the edge grammars for a graph family under which the family can be shown to be truncatable.

Definition: Let $\{G(n)=(V(n),E(n))\}$ be a graph family. Then $\{G(n)\}$ is *contractable* if for each n, there is a mapping $c:V(n+1)\rightarrow V(n)$ such that $c(V(n+1))\subseteq V(n)$ and $\{(c(v),c(w))|\ (v,w)\varepsilon E(n+1)\}\subseteq E(n)$.

Definition: Let $G=(V,E)$ be a graph whose labels are strings in Σ^*. Let m be the mapping which assigns to each label xa $(x\varepsilon\Sigma^*,\ a\varepsilon\Sigma)$ in V the label x in Σ^*. Then the graph $m(G)=(\{m(v)|\ v\varepsilon V\},\{(m(v),m(w))|(v,w)\varepsilon E\})$ is said to be a *truncation* of G.

Definition: A graph family $\{G(n)\}$ is *truncatable* if for each $n>0$, $G(n)$ is a truncation of $G(n+1)$.

Proposition

If a graph family is truncatable, it is contractable.

The proposition follows directly from the definitions. Note however that not all contractable graph families are truncatable. The family of shuffle-exchange graphs {SE(n)} with the usual labeling (i.e. if vertex w is connected to vertex v by a shuffle or exchange edge then the label of vertex w is the shuffle or exchange of the label of vertex v) is not truncatable since the truncation mapping for this labeling does not preserve adjacencies. (For example, shuffle edge (bxa,xab) in SE(n) is truncated to the edge (bx,xa) in SE(n-1) which is neither a shuffle nor an exchange edge when $b\neq a$). However the shuffle-exchange family is contractable with the trivial contraction which maps every vertex in SE(n) onto a single vertex in SE(j) $1\leq j<n$.

Theorem 2

Let Γ be a Type 3 edge grammar. Assume that

1) *For each (v,w) in T, $|v|=|w|=1$,*
2) *For each nonterminal A in N-$\{G\}$, there is a production $G\rightarrow A$ where G is the start symbol.*

Then $\{G(n)\}$ is truncatable.

Proof

Let $G\rightarrow{}^*(v,w)$ with $|v|=|w|=n+1$. Then $v,w\varepsilon V(n+1)$ and if $v\neq w$, $(v,w)\varepsilon E(n+1)$. Since Γ is *Type 3*, there is a nonterminal H and a terminal pair (a,b) such that $G\rightarrow H(a,b)\rightarrow{}^*(v,w)$. By property 1), $|a|=|b|=1$ and by property 2, $G\rightarrow H$. Hence there are strings x, y with $v=xa$, $w=ya$, $|x|=|y|=n$ and $G\rightarrow H\rightarrow{}^*(x,y)$. In particular, if $x\neq y$, then (x,y) is in $E(n)$. In any case, x and y are in $V(n)$. Hence $G(n)$ is a truncation of $G(n+1)$.

Note that embedding by truncation may still achieve good results even if Theorem 2 is not applicable. The truncation mapping from SE(n) (the shuffle-exchange graph with 2^n nodes) into SE(j) for $1<j\leq n$ maps label $x_1 \ldots x_n$ onto $x_1 \ldots x_j$. The exchange edges $(x_1 \ldots x_{n-1}0, x_1 \ldots x_{n-1}1)$ are mapped to the single vertex $x_1 \ldots x_j$, and the shuffle edges $(x_1 \ldots x_{n-1}a, ax_1 \ldots x_{n-1})$ are mapped onto $(x_1 \ldots x_j, ax_1 \ldots x_{j-1})$. Note that if $a=x_j$, this is already a shuffle edge in SE(j) and if $a\neq x_j$, there is a path of length 2: $(x_1 \ldots x_j, x_1 \ldots x_{j-1}a)$, $(x_1 \ldots x_{j-1}a, ax_1 \ldots x_{j-1})$ (exchange edge, shuffle edge) into which $(x_1 \ldots x_j, ax_1 \ldots x_{j-1})$ can be embedded in SE(j). Since no truncation of SE(n) with the usual vertex labeling is adjacency-preserving, this worst-case expansion of at most 2 for the edges of SE(n) is the best we can do in this case. Further, after truncation, the vertices of SE(n) are distributed optimally over the vertices of SE(j) since each vertex in SE(j) is the image of 2^{n-j} vertices in SE(n).

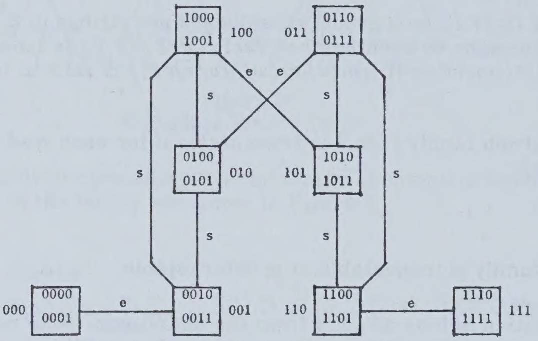


Figure 4
Embedding the truncation of SE(4) into SE(3).

SECTION 3: EDGE GRAMMARS AND PARALLEL COMPUTATION

Given a parallel algorithm to be run on a specific parallel computation device, the implementation of the algorithm on the device can be abstracted as an embedding of the interconnection graph for the algorithm into the specific interconnection structure for the device. Since the parallel algorithm represents a set of problem instances, its proper abstraction is a graph family, each member of which represents the interconnection structure of a specific problem size. In embedding a given interconnection structure into a parallel device, it is typically assumed that there will be enough processors on the device so that the interconnection graph can be laid out with an assignment of one processor per node.

In practice, this is frequently not the case. For large problem sizes, the number of nodes in the interconnection graph may exceed the number of processors in the given parallel device. Hence an implementation of a parallel algorithm on a parallel computer may involve finding an embedding of the large interconnection graph into the interconnection structure of the given device. This can always be done in a trivial way by mapping the entire interconnection graph into a single processor, but this solution is neither desirable nor efficient since the embedding would have poor performance. The goal is to embed the interconnection graph into the device so that the processor usage is balanced and so that the algorithm can be multiplexed efficiently on the given architecture.

Theorem 2 provides an effective procedure by which we can show many graph families to be truncatable. For parallel algorithms whose graph families are truncatable, there is a uniform algorithm for mapping large graphs from the family into a fixed size parallel architecture. The algorithm is

1) *Choose an interconnection graph G' from the family which can be laid out efficiently on the interconnection structure H of the given parallel device. Assume the length of the labels in G' is n'.*
2) *For a large problem size n of the algorithm, truncate the labels of the interconnection graph to the first n' bits. By definition of truncation, the resulting graph is a subgraph of G'.*
3) *Embed G' in H and multiplex G on G' by substituting for each node in G its image under the truncation.*

For truncatable graph families, truncation is adjacency-preserving, hence any layout of *G'* on *H* will accommodate *G*. An important feature of this strategy is that the layout problem can be separated from the embedding problem. In particular, solutions to both problems can be optimized.

As an example, suppose we wish to execute a parallel algorithm whose interconnection graph is a binary tree with 255 processors and we have available a parallel machine with 64 processors configured as a cube-connected cycle [P1]. For example, the algorithm could be the Schwartz's parallel max-finding algorithm [S4] or an algorithm written for Browning's Tree Machine [B]. By Theorem 2, the family of complete binary trees is truncatable, hence we can map the 255 node tree onto the 63 node binary tree, and lay the 63 node tree out on the cube-connected cycle using an embedding (Figure 6) which maps each edge of the tree into a path of length at most 2 on the cycle. The algorithm is then multiplexed by replacing references to processor $a_1 \ldots a_5 a_6 a_7$ by references to processor $a_1 \ldots a_5$. In general, the truncation from $G(n+1)$ to $G(n)$ "folds" the left subtree onto the right subtree and identifies the root, left son and right son of $G(n+1)$ with the root of $G(n)$. We do this contraction twice to obtain the 63 node tree from the 255 node tree as shown in Figure 5.

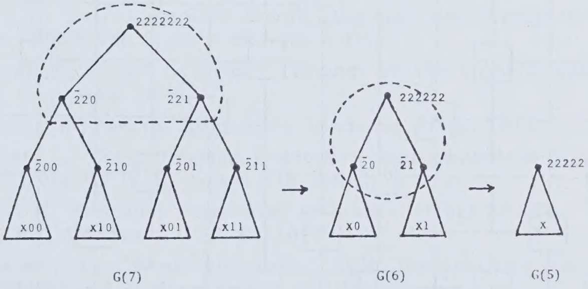


Figure 5
Contraction of the 255 node tree into the 63 node tree.

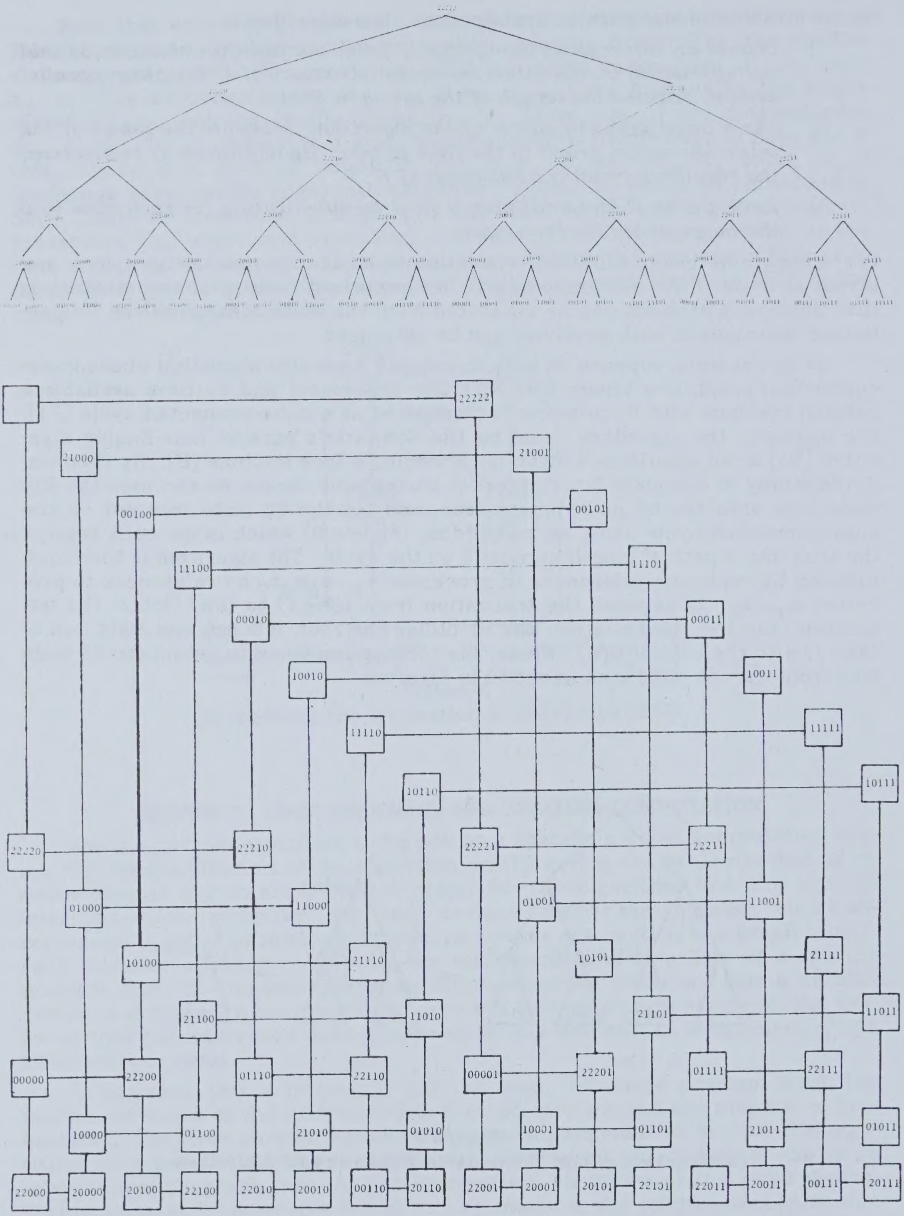

22222
22220
22221
22200
22210
22201
22211
22000
22100
22010
22110
22001
22101
22011
22111
20000
21000
20100
21100
20010
21010
20110
21110
20001
21001
20101
21101
20011
21011
20111
21111
00000
10000
01000
11000
00100
10100
01100
11100
00010
10010
01010
11010
00110
10110
01110
11110
00001
10001
01001
11001
00101
10101
01101
11101
00011
10011
01011
11011
00111
10111
01111
11111
22222
21000
21001
00100
00101
11100
11101
00010
00011
10010
10011
11110
11111
10110
10111
22220
22210
22221
22211
01000
11000
01001
11001
10100
21110
10101
21111
21100
11010
21101
11011
00000
22200
01110
22110
00001
22201
01111
22111
10000
01100
21010
01010
10001
01101
21011
01011
22000
20000
20100
22100
22010
20010
00110
20110
22001
20001
20101
22101
22011
20011
00111
20111

Figure 6
Layout of the 63 node tree on the 64 processor cube-connected cycle.

Acknowledgements

It is a pleasure to thank Larry Snyder for many helpful discussions and suggestions. His thoughtful comments have contributed materially to both the content and the organization of this report. In addition, Mike Atallah provided many helpful comments and suggestions on an earlier draft of this paper.

REFERENCES

[AHU] Aho, A., Hopcroft, J. and J. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, 1975.

[B] Browning, S., "The Tree Machine: A Highly Concurrent Programming Environment," Ph.D. Thesis, California Institute of Technology, 1980.

[CER] Claus, V., Ehrig, H. and G. Rozenberg, Lecture Notes in Computer Science 73: *Graph Grammars and Their Application to Computer Science and Biology*, Springer-Verlag, 1979.

[LP] Lewis, H. and C. Papadimitriou, *Elements of the Theory of Computation*, Prentice-Hall, 1981.

[NS] Nassimi, D. and S. Sahni, "Data Broadcasting in SIMD Computers, 1" IEEE Transactions on Computers, C-30, #2, 1981.

[P1] Preparata, F., "Algorithm Design and VLSI Architecture," University of Illinois Coordinated Science Lab Tech. Report, 1982.

[P2] Pratt, T., "Pair Grammars, Graph Languages and String-to-Graph Translations," JCSS, Vol. 5, #6, p. 560-595, 1971.

[RS] Rosenberg, A. and L. Snyder, "Bounds on the Costs of Data Encodings," MST, 12, p. 9-39, 1978.

[S1] Salomaa, A., *Formal Languages*, Academic Press, 1973.

[S2] Snyder, L., "Programming Processor Interconnection Structures," Purdue University Tech. Report #381, 1981.

[S3] Stone, H., "Parallel Processing with the Perfect Shuffle," IEEE Transactions on Computers, C-20, #2, 1971.

[S4] Schwartz, J., "Ultracomputers," ACM Transactions on Programming Languages and Systems, Vol. 2, #4, 1980.

[V] Valiant, L., "Universality Considerations in VLSI Circuits," IEEE Transactions on Computers, C-30, #2, 1981.

GRAPH SEPARATION AND SEARCH NUMBER

J.A Ellis[(1*)], I.H Sudborough[(1*)], J.S Turner[(2)]
(1) Electrical Engineering and Computer Science
Department, Northwestern University, Evanston, Ill 60201
(2) Computer Science Department
Washington University, St. Louis, Missouri 63130

ABSTRACT

We relate two different concepts in graph theory and algorithmic complexity, namely "searching" a graph [1] [2] and "separators" of undirected graphs [3] [4] [5]. For any graph G, let s(G) denote the "search number" of G and vs(G) denote the vertex separation of G. We show that $vs(G) \leq s(G) \leq vs(G) + 2$ and that if G' is created from G by placing two added vertices in every edge in G, then $vs(G') = s(G)$.
We discuss algorithms which show that for all fixed $k \geq 1$ and graphs G the following problems are decidable in polynomial time: "Is $vs(G) \leq k$?", "Is $s(G) \leq k$?", "Is the progressive black/white pebble demand of $G \leq k$?". When both G and k are input quantities, these problems are NP complete [6] [7].
We also give an algorithm that for any tree T computes vs(T) in $O(n \log n)$ steps, where n is the number of vertices in the tree, describe the smallest trees with a given vertex separation and characterize trees of a given vertex separation.

1. INTRODUCTION

This paper is an abbreviated version of a paper in preparation which contains detailed proofs of the theorems stated here. Searching a graph is a concept introduced by Parsons [1]. Informally, the search number of a graph G, denoted by s(G), is the minimum number of searchers necessary to guarantee the capture of a fugitive who can move with arbitrary speed about the edges of the graph and who has perfect knowledge of the movements of the searchers. Meggido et al [2] show that determining the search number of a graph is NP-hard and that the search number of a tree can be determined in O(n) steps. It is known that, for any graph G with maximum vertex degree 3, s(G) is identical to the "cutwidth" of G [8]. Hence the search number problem has practical value since finding the cutwidth of a graph is important in some VLSI layout applications [3].

A search sequence is a sequence of the following steps: (a) place a searcher on a vertex, (b) slide a searcher through an edge and (c) remove a searcher from a vertex. We say that an edge $e=\{x,y\}$ is cleared if either (1) there is a searcher on x and a second searcher is moved from x to y or (2) there is a searcher on x, all edges incident to x except e have been cleared, and the searcher on x is shifted through e to y. In the beginning all edges are "contaminated", i.e. not cleared. Searching a graph means to reach a state in which all edges

---

[*] Research supported in part by NSF grant #MCS81-09280

are simultaneously cleared. A cleared edge e can become recontaminated by the movement or deletion of a searcher which results in a path without searchers from a contaminated edge to e. LaPaugh [7] has shown that there is always an optimal, so called progressive, search sequence which never allows any edge to be recontaminated. Hence the search number problem is NP complete.

Separation of undirected graphs is an important concern in complexity theory. A separator of an undirected graph is a set of vertices, in the case of a vertex separator, or a set of edges, in the case of an edge separator, whose removal separates the graph into two components. Separators are used to describe good VLSI layouts in [3] and have been used to describe good divide and conquer algorithms in [4]. Separator theorems for planar graphs have been described by Lipton and Tarjan [5]. Lengauer [6] called the above definition of separator static and goes on to define a dynamic "vertex separator game". We consider the same concept as Lengauer but describe it in terms of linear layouts.

Let $G=(V,E)$ be a graph. A linear layout, or simply a layout, of G is a one-one mapping $L:V \dashrightarrow \{1,2,\ldots,|V|\}$.
Define $V_L(i) = \{u \text{ in } V \mid \{u,v\} \text{ in } E \text{ and } L(u) \leq i < L(v)\}$.
Vertices in $V_L(i)$ are called the active vertices at time i. The vertex separation of G with respect to L, denoted by $vs_L(G)$, is defined by

$$vs_L(G) = \max_{1 \leq i < n} (|V_L(i)|)$$

and the vertex separation of G is defined by

$$vs(G) = \min_L (vs_L(G))$$

Examples of layouts which minimize vertex separation are given in Figures 2.1 and 2.2.

2. RELATIONSHIPS BETWEEN VERTEX SEPARATION AND SEARCH NUMBER

Theorem 2.1

Let $G = (V,E)$ be a graph. Then $vs(G) \leq s(G) \leq vs(G) + 2$.

Proof Sketch To show that $vs(G) \leq s(G)$ we show that given a search sequence we can use it to derive a layout. Let S be a progressive search sequence for G. We define the layout L for G by assigning the vertex x to the integer i if it is the i-th vertex to be first visited by a searcher in the sequence S. It can be shown that, since S is a progressive strategy and does not allow recontamination, there must be at least a searcher on every active vertex at each point in the layout. Hence $vs(G) \leq s(G)$.

To show that $s(G) \leq vs(G) + 2$ we show that given a layout we can use it to derive a search strategy. Let L be a layout for G. We clear all the edges of G by the sequence of steps given in the procedure search described below. It is easily seen that the procedure is effective and that it never uses more than (the number of currently active vertices + 2) searchers .

Hence $s(G) \leq vs(G) + 2$.

```
procedure search(G,L);
    for i := 1 to |V| do
        begin x := $L^{-1}(i)$; Place a searcher on x;
        for each left neighbor y of x do
            begin add a new searcher to y;
            slide a searcher from y to x;
            remove a searcher from x
            end;
        Remove searchers from inactive vertices in $L_i$
        end;
```

The bound in Theorem 2.1 is the best possible because the graph $K_{3,3}$ shown in Figure 2.1 has vertex separation three and search number five.

Let the 2-expansion of a graph G be the graph formed by replacing each edge $\{x,y\}$ of G with the two new vertices, say u,v, and the edges $\{x,u\}$, $\{u,v\}$, and $\{v,y\}$. For example, Figures 2.2 and 2.3 show a graph G and its 2-expansion.

Theorem 2.2. For any graph G, s(G) is identical to the vertex separation of the 2-expansion of G.

Proof Sketch In Theorem 2.1 we have seen that, for any graph G, $vs(G) \leq s(G)$. Let G´ be the 2-expansion of G. Clearly, subdividing edges does not change search number, so $s(G) = s(G')$. Thus, it follows that $vs(G') \leq s(G)$.

To show that $s(G) \leq vs(G')$ we argue similarly as in the proof of Theorem 2.1. Let L be a layout of G´ such that $vs_L(G')=k$. It can be shown that there is a layout of G´ with vertex separation k such that, for any edge $e = \{x,y\}$ in G, the vertices added to e are given consecutive positions and are laid out either to the right of x or to the right of y. We construct a searching algorithm based on a layout L which does satisfy these properties. Because of the special property there is now no need to use more searchers than the number of active vertices at each point. Figure 2.2 shows the complete graph on four vertices which has search number 4 and vertex separation 3. Figure 2.3 shows the 2-expansion and a layout satisfying the properties just described and with a vertex separation of 4, which is optimal. The search number is unchanged.

3. THE VERTEX SEPARATION OF TREES

Trees allow the possibility of computing certain properties of the root by recursively computing the property for its subtrees and then combining the results. For example, Meggido et al [2] give an algorithm for computing the search number of a tree, Chung et al [9] give an algorithm for computing the cutwidth of a tree, and Yannakis [10] describes an improved cutwidth algorithm for trees which is also effective for computing their black/white pebble demand.

Our algorithm also is based on a recursive characterization of the vertex separation of a tree in terms of the vertex separation of the subtrees induced by its root. The algorithm is easily developed to give an optimal layout for vertex separation for the given tree.

3.1 A Recursive Characterization of Trees with Vertex Separation k

The subtrees induced by a vertex x are all of the trees in the forest obtained by deleting x from the given tree. A tree with one vertex has vertex separation 0 and we shall adopt the convention that the empty tree has vertex separation -1.

Theorem 3.1 Let T be a tree. For any integer $k \geq 1$, let P(k) denote the following property of T:
For all vertices x at most two subtrees induced by x have vertex separation k and all remaining subtrees induced by x have vertex separation at most k-1.
T has vertex separation at most k if and only if P(k) is true.

Proof Sketch We first show that if T satisfies P(k), then there is a layout L of T such that $vs_L(T) \leq k$. Let V_k be the set of vertices in T which induce exactly two subtrees each with vertex separation k. It can be shown that there exists a path containing all the members of V_k. Let $x_1, x_2, .. , x_p$, for some $p \geq 1$, be a chain of vertices such that, for all i ($1 \leq i \leq p$), x_i induces subtrees which, if they do not contain other chain vertices, have vertex separation at most k-1. Given such a chain we can construct a layout with vertex separation k by laying out all the subtrees with vertex separation $\leq$ k-1 in sequence immediately to the right of the chain vertex that induces them. See Figure 3.1. In this figure a, b, c and d are members of the chain.

Now we show that if there is a layout L such that $vs_L(T)$ is at most k, then P(k) is true. Let vertices a and b be the first and last vertices in such a layout. Let x be any vertex in T. There must be paths from both a and b to x. Remove from the layout the vertex x, all edges incident to x, and the one or two subtrees induced by x and containing the vertices a and b. It can be seen that for all remaining subtrees T', $vs(T') \leq k-1$ because the paths from a and b to x are removed. No more than two subtrees were removed and these had vertex separation $\leq$ k.

Corollary 3.1 vs(T) > k iff there exists a vertex which induces 3 subtrees T' such that $vs(T') \geq k$.

For example, the tree shown in Figure 3.2 has vertex separation 3 and not 2, since the indicated vertex x induces three subtrees with vertex separation 2.

3.2 k-Criticality

In the following we shall consider directed trees for convenience. The vertex separation of a directed tree is the vertex separation of the underlying undirected tree. Let $T<x>$ denote the subtree of the directed tree T with root vertex x.

Definition 3.1. A vertex x is *k-critical* in a directed tree T if (1) $vs(T<x>) = k$ and (2) there are two children y and z of x such that $vs(T<y>) = k$ and $vs(T<z>) = k$.

Let $T<x,v_1, v_2, \ldots v_i>$ denote the subtree of T with the root vertex x from which the subtrees $T<v_1>, \ldots , T<v_i>$ have been deleted. It follows from Theorem 3.1 that, if T^i is a directed tree with root x and vertex separation k, and there is a k-critical vertex v in T, then $vs(T<x,v>) \leq k-1$. We shall assign a set of integers as a label to all vertices in a tree. These labeling sets have the following interpretation. If a vertex x is assigned the label $(a_1, a_2, \ldots, a_d)$ in the directed tree T, then:

1. $vs(T<x>) = a_1$,
2. for all i ($1 \leq i < d$), there exists an a_i-critical vertex v_i such that $vs(T<x,v_1,v_2,\ldots,v_{i-1},v_i>) = a_{i+1}$,
3. $vs(T<x,v_1,v_2,\ldots,v_{d-1}>) = a_d$ and $T<x,v_1,v_2,\ldots,v_{d-1}>$ does not have an a_d-critical vertex.

For example the label $\{2,0\}$ on vertex x means that $T<x>$ has vertex separation 2, that there is a 2-critical vertex v and that $T<x,v>$ has vertex separation 0. The label $\{2,-1\}$ on the vertex x means that the vertex separation of T(x) is 2 and the vertex v=x is a 2-critical vertex so that $T<x,v>$ is empty.

We now describe a procedure for combining the labels of the d subtrees induced by some vertex x to compute the label of x.

```
procedure compute-label($S_1,S_2,\ldots,S_d$)
k <--- max( $S_1 \cup S_2 \cup \ldots \cup S_d$ );
if k = 0 then return {1};
if d = 1
then begin
      if min $S_1$ > 0 then return $S_1$;
      if min $S_1$ = -1 then return {0} U {i > 0 | i is in $S_1$}
      i <--- min { j | j > 0 and j is not in $S_1$ };
      return {i} U { j > i | j is in $S_1$ }
      end;
m <--- | { $S_i$ | k is in $S_i$ } |;
if m $\geq$ 3 then return {k+1};
if m=2
then begin (* let k be in the sets $S_x$ and $S_y$ *)
        if $S_x$ > {k} or $S_y$ > {k} then return {k+1};
        return {k,-1}
         end;
 (* In this case the element k is in one set, say $S_x$ *)
if k = min $S_x$ then return {k}
H <--- compute-label($S_1,S_2,\ldots,S_x-\{k\},\ldots,S_d$);
if H $\geq$ {k} then return {k+1} else return {k} U H
```

An outer procedure, called initially on the root, will return {0} as the label all leaves. On other than leaves it recursively calls itself on each of the children and passes the resulting labels to the procedure compute-label which computes the label of x. The vertex separation of a tree is then the largest number in the label of its root.

The correctness of the procedure can be established by considering each of the various cases that it handles and applying Theorem 3.1. An analogous procedure for computing the cutwidth of trees was described in [9].

Assume that suitable data structures are used and note that there are O(log n) entries in each label, because (section 3.4) the maximum vertex separation of a tree with n vertices is O(log n). Recursion depth is limited by the number of items in a label so the time complexity of the procedure combine-label is $O(d_i * \log n)$, where d_i is the degree of the ith vertex. The procedure is invoked for all vertices in the tree. Since the sum of the vertex degrees over all vertices is O(n) we have that the time complexity of the entire process is O(n log n).

3.3 Computing an Optimal Layout

Once labels have been computed for all vertices, an optimal layout can be computed. This is accomplished by the following procedure in which label(x) means the label of x, pos is the position assigned to a vertex and L(x) is the layout function.

```
procedure layout (x);
    k <-- max label(x); c <-- x;
    while c is not a k-critical vertex do
    begin delete k from label(c);
        c <-- the child of c with k in label(child)
    end;
    Let $(v_1, v_2, ..... v_s)$ be the path of all
    vertices in T<c> with k in the label;
    for i := 1 to s do
        begin L($v_i$) <-- pos;
        pos <-- pos + 1; delete $v_i$ from T;
        for all children y of $v_i$ do layout(y);
        if $v_i$ has a parent not in $(v_1 ... v_s)$
        then layout(x)
        end
```

The correctness of this procedure can be demonstrated by noting that it is finding the chain of vertices that induce subtrees of vertex separation k-1 or less, as discussed in the proof of Theorem 3.1, and laying out these subtrees to the right of their inducing chain vertex.

The time complexity of the procedure is O(n log n) since no vertex is visited more than k times and k is O(log n) as shown in section 3.4.

3.4 Smallest Trees with a Given Vertex separation

Let T(k) denote the collection of trees with the smallest number of of vertices and vertex separation k. It is easy to see that T(1) and T(2) contain just one tree. There are several trees in the set T(3). One of is shown in Figure 3.2. To construct a tree in the set T(k+1), for all k, one takes three trees from T(k), say T_1, T_2, and T_3, which need not be distinct, and a new vertex x, and joins the three subtrees by adding an edge from an arbitrary vertex of T_1, T_2, and T_3 to the vertex x. It follows from Theorem 3.1 that the constructed tree has vertex separation k+1. Furthermore, from Theorem 3.1, any tree with vertex separation k+1 must have a vertex x such that x induces at least three subtrees with vertex separation at least k. Consequently, the constructed tree must be among the smallest trees with vertex separation k+1.

One can also deduce that the number of vertices in a smallest tree with vertex separation k is $\lfloor 5/6*3^k \rfloor$ for all $k \geq 0$ so that vs(T) is O(log $|V|$).

3.5 Characterization of Trees with Vertex separation k

The operation of replacing an edge $\{x,y\}$ with a new vertex z and the two edges $\{x,z\}$ and $\{z,y\}$ is called edge subdivision. A graph G' is a homeomorphic image of a graph G when G' can be obtained from G by a finite number of edge subdivisions.

Let T be a tree. Let S(T) denote the set of all trees that can be obtained from T by a single edge subdivision operation. Similarly, if F is a family of trees, then S(F) denotes the family of trees $\{$ S(T) $|$ T is in F $\}$. For all $i \geq 1$, let F(i) be the family of trees defined by:
F(1) = T(1); F(i+1) = the collection of all trees that can be formed by taking three trees in F(i) U S(F(i)) and a new vertex x and joining x by an edge to an arbitrary vertex in each tree, for all $i \geq 1$.

Theorem 3.2. For all $k \geq 1$, a tree has vertex separation at least k if and only if it contains a subtree which is a homeomorphic image of a tree in F(k).

A proof by induction on the size of the tree is straightforward. A corollary follows immediately.

Corollary 3.1. For all $k \geq 1$, a tree has vertex separation k if and only if it contains a subtree that is a homeomorphic image of a tree in F(k) and does not contain a subtree which is a homeomorphic image of a tree in F(k+1).

4. RECOGNIZING GRAPHS WIH FIXED VERTEX SEPARATION

In [11] [12] and [8] dynamic programming algorithms have been described which recognize graphs with fixed bandwidth or cutwidth in polynomial time. However,the application of these earlier algorithms to the problem of vertex separation does not produce a polynomial time algorithm because no vertex degree constraint follows from the fact that a graph has a small vertex separation. For example, the star graph S_n, which is the tree with n leaves and a single internal vertex with degree n, has vertex separation 1 for all n. Consequently, if one defines an equivalence relation on the class of partial layouts as was done in the algorithms just cited, then the number of equivalence classes is not bounded by any polynomial in the number of vertices. A dynamic programming algorithm has been devised for computing vertex separation which uses an equivalence relation for vertex separation k that divides the class of all plausible partial layouts into $O(n^m)$ equivalence classes, where $m = (k^2+5k+2)/2$. Since the method requires that each class be considered O(n) times, the following Theorems can be derived.

Theorem 4.1
For each $k \geq 1$ there is an $O(n^m)$ step algorithm, where $m = (k^2+5k+4)/2$, to decide if a graph with n vertices has vertex separation k.

Since we can expand any graph and the vertex separation of the result is equal to the search number of the original graph, we have the following, which solves an open problem in Meggido et al [2]. The expansion was illustratd in Figure 2.3.

Corollary 4.1
For each $k \geq 1$ there is an $O(n^m)$ step algorithm, where $m = k^2+5k+4$, to decide if a graph G with n vertices has search number k.

The transformation described by Lengauer [6] from the black-white pebble game to the vertex separation problem justifies the following corollary.

Corollary 4.2
For each $k \geq 1$ there is an $O(n^m)$ step algorithm, where $m = (k^2+5k+4)/2$, to decide if a graph G with n vertices has pebble demand k in the progressive black-white pebble game.

REFERENCES

[1] Parsons, T. D, "Pursuit-Evasion in a Graph", in Theory and Application of Graphs, Y. Alavi and D. R. Lick,(eds) Springer-Verlag, Berlin, 1976, pp. 426-441

[2] Meggido, N, Hakimi,S. L, Garey M. R, Johnson, D. S and Papadimitriou, C. H, "The Complexity of Searching a Graph (Preliminary Version)", Proc. IEEE Foundations of Computer Science Symp. (1981), pp. 376-385

[3] Leiserson, C. E, "Area-Efficient Graph Layouts, for VLSI", 21st Annual IEEE Symposium on FOCS, (1980)

[4] Lipton, R. J and Tarjan, R. E, "Applications of a Planar Separator Theorem", SIAM J. Comput. 9,3 (1980), pp. 615-627

[5] Lipton, R. J and Tarjan, R. E, "A Separator Theorem for Planar Graphs", SIAM J. Appl. Math. 36,2 (1979), pp. 177-189

[6] Lengauer, T, "Black-White Pebbles and Graph Separation", SIAM J. Algebraic and Discrete Methods, 1982.

[7] LaPaugh, A. S., "Recontamination does not Help to Search a Graph", Technical Report, Electrical Engineering and Computer Science Department, Princeton University (1983)

[8] Makedon, F. and Sudborough, I. H, "Minimizing Width in Linear Layouts", Proc. 10th International Colloquium on Automata, Languages, and Programming, vol. 154, Lecture Notes in Computer Science, Springer Verlag (1983), pp. 478-490.

[9] Chung, M. J, Makedon, F, Sudborough, I. H and Turner, J, "Polynomial Algorithms for the Min-Cut Linear Arrangement Problem on Degree Restricted Trees", Proc. 23rd Annual IEEE Foundations of Computer Science Symp. (1982), pp. 262-271.

[10] Yannakakis, M., "A Polynomial Algorithm for the Min Cut Linear Arrangement of Trees", to appear in Proc. 24th Annual IEEE Foundations of Computer Science Symposium (1983)

[11] Saxe, J. B, "Dynamic-Programming Algorithms for Recognizing Small Bandwidth Graphs in Polynomial Time", SIAM J. on Algebraic and Discrete Methods, December, 1980

[12] Gurari, E. M and Sudborough, I. H., "Improved Dynamic Programming Algorithms for Bandwidth Minimization and the Min-Cut Linear Arrangement Problem", J. Algorithms, to appear.

FIGURES

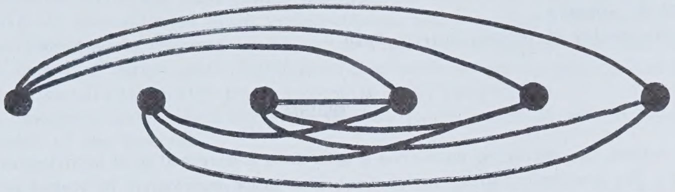


Figure 2.1 The Graph $K_{3,3}$ with vs(G)=3 and s(G)=5

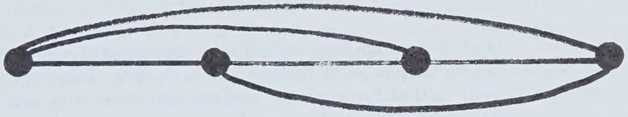


Figure 2.2 A Graph with vs(G)=3 and s(G)=4

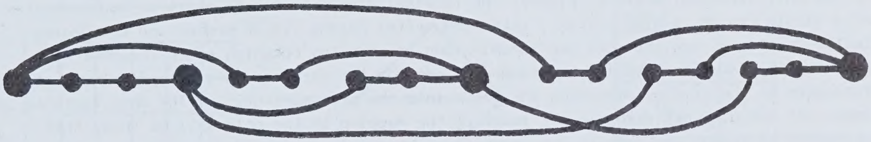


Figure 2.3 The Expanded Graph from 2.2 with vs(G′)=s(G)=4

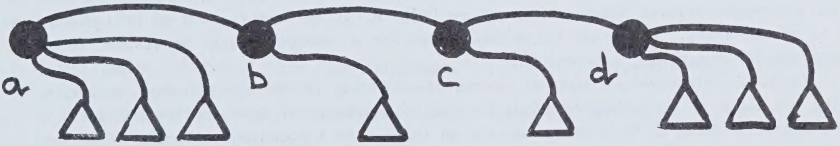


Figure 3.1 Inserting Subtrees in the Chain so that vs(T)=k

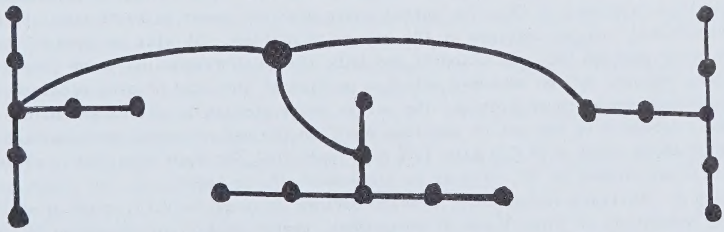


Figure 3.2 One of the Smallest Trees with vs(T)=3

MINIMAL AREA GRAPH EMBEDDING INTO A DIRECTED GRID IS NP-HARD

HILDA M. STANDLEY
The University of Toledo, Toledo, Ohio

ABSTRACT

A scheme for embedding data flow graphs into a directed grid architecture is presented. The problem of obtaining minimal area graph embeddings is stated in a theorem to be NP-hard. An outline of the proof is given.

SUMMARY

Recent advances in VLSI technology have made the processor-grid a practical alternative for an asynchronous computer architecture [3, 5, 7, 8]. This type of architecture is especially well suited for a direct implementation of data driven computations as represented by data flow graphs [1, 2], allowing much of the asynchronous nature of data flow to be preserved during execution.

The planar directed processor-grid architecture with identical processors located at each vertex is used to accommodate a restricted class of data flow graphs chosen to be representative of generalized data flow graphs [6]. This class of graphs is defined to include only directed, acyclic, planar, two-terminal graphs (a single start or source vertex and a single finish or sink vertex), with each vertex having its in-degree and out-degree limited to at most two (at most two incoming and at most two outgoing edges incident to a vertex). This architecture is programmed by mapping the vertices in a data flow graph to processors in the grid by embedding the graph into the processor-grid. The only vertices in common for two distinct paths in the image of the mapping in the grid will be those that correspond to common vertices in the graph (i.e. cross-overs not permitted). Figure 1 illustrates the directed grid.

Two major concerns in the implementation of this general-purpose architecture are the identification of the class of graphs that may be embedded and obtaining area-efficient and time-efficient embeddings. The feasibility of any graph of this class being embedded into the directed processor-grid is addressed elsewhere [6]. A further restriction on this class was found to be necessary. Any graph being considered for an embedding will be assumed to have been previously determined "embeddable."

The area, expressed in units of processor-vertices, of the minimal-area rectangle bounding the image of the embedded graph is used as a measure of resource consumption. A minimal area embedding is then one producing an image with a bounding rectangle of minimal area. The area-efficient embedding problem may be stated as follows: given an embeddable graph, find the graph embedding into the processor-grid of minimal area. The embedding problem (EP) will be shown to be NP-hard.

In order to prove EP is NP-hard, it may be shown that an NP-complete problem may be transformed to an instance of EP. The vertex cover problem, known to be NP-complete [4], will be transformed into an instance of the embedding problem. It will be demonstrated that the vertex cover problem has a solution if and only if the corresponding graph (instance of the embedding problem) may be embedded within a portion of the grid of area determined by two parameters of the vertex cover problem. The vertex cover problem is stated as follows. Given graph $G=(V,E)$ (where V is the set of vertices and E is the set of edges) and positive integer $K \leq |V|$, does there exist a $V^o \subseteq V$ with $|V^o| \leq K$ such that for each edge (u,v), element of E, either u is an element of V^o, or v is an element of V^o, or both?

Given an arbitrary vertex cover problem defined on graph $G=(V,E)$, graph g will be constructed consisting of three types of subgraphs: vertex units (corresponding to elements of V), edge units (corresponding to elements of E), and blank units (corresponding to pairs of vertices having no edge between them). The edge units of g are constructed so that an embedded graph whose side length is determined only by the number of edges along a path in the edge unit would produce a conflict (the assignment of more than one graph vertex to a grid vertex). As an example of the construction, consider the graph given in figure 2.

Figure 3 shows the relative positions of the units of the corresponding constructed directed graph, g, where vertex units are labeled with the corresponding vertex labels, edge units are labeled with pairs of vertex labels given in parentheses, and blank units are labeled B. Vertex units corresponding to vertices in the vertex cover V^o will be embedded into larger portions of the grid than vertex units corresponding to vertices not included in the cover. This increase in area will be limited by an expression in $|V^o|$ and K.

The two problems (vertex cover and graph embedding) may now be related as shown in the following statement of the theorem.

Theorem:

Given graph $G=(V,E)$, positive integer $K \leq |V|$, and directed graph g constructed from graph G, vertex cover $V^o \subseteq V$ with $|V^o| \leq K$ exists if and only if g may be embedded within grid square of side length $|V| * (2 * (|V| - 2) + K + 7) + K$.

For the necessity part of the proof, the additional side length of K, guaranteed by the existence of vertex cover V^o with $|V^o| \leq K$, gives a large enough increase in the area of the embedded graph to allow the resolutions of all edge unit conflicts. For the sufficiency part of the proof, the resolution of all edge unit conflicts in a successful embedding within the given grid area guarantees the existence of a vertex cover of size no greater than K. In conclusion, the embedding problem is NP-hard and is, therefore, at least as "hard" as any NP-complete problem.

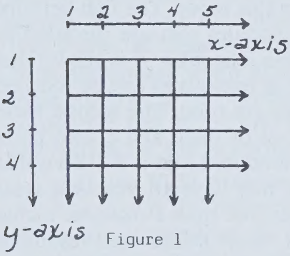


Figure 1

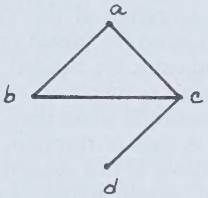


Figure 2

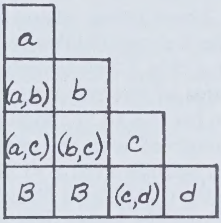


Figure 3

Bibliography

[1] William B. Ackerman, "Data Flow Languages," AFIPS Conf. Proceedings, pp. 1087-1095, 1979.

[2] Tilak Agerwala, Arvind, "Data Flow Systems," Computer, Vol. 15, No. 2, pp. 10-13, Feb. 1982.

[3] Douglas G. Fairbairn, "VLSI: A New Frontier for Systems Designers," Computer, Vol. 15, No. 1, pp. 87-96, Jan. 1982.

[4] R. M. Karp, "Reducibility Among Combinatorial Problems," in Complexity of Computer Computations, pp. 85-103, 1972.

[5] Charles E. Leiserson, "Area-Efficient Graph Layouts (for VLSI)," IEEE 21st Annual Symposium on Foundations of Computer Science, pp. 270-281, 1980.

[6] Hilda M. Standley, "Data Flow Graph Embeddings Into a Directed Processor Grid," Ph.D. Dissertation, The University of Toledo, March 1983.

[7] C. D. Thompson, "Area-Time Complexity for VLSI," Proc. of the 11th Annual ACM Symposium of Theory of Computing, pp. 81-88, 1979.

[8] Leslie G. Valiant, "Universality Considerations in VLSI Circuits," IEEE Trans. on Computers, Vol. C-30, No. 2, pp. 135-140, Feb. 1981.

On I/O Placement in VLSI Circuits
(Extended Abstract)

Alok Aggarwal

Electrical Engineering and Computer Science
The Johns Hopkins University
Baltimore, MD 21218.

Introduction

Theoretical aspects of VLSI computation have garnered considerable attention in recent years. In particular, effort has been spent in establishing area-time tradeoffs for various functions [Th79, Th80, Vu80, Sa80, BK81]. Also, motivated by both the practical necessity and the theoretical importance, researchers have identified bounds for different kinds of I/O schedules [LS81, Sa81, Ja81, KZ81]. However, there has been little progress in determining the effect of different I/O placements on the area-time complexity of any function.

Three issues related to I/O placement are considered in this paper. The first pertains to the incompatiblities in the I/O schedules of the VLSI modules that are cascaded to form a chip. Close bounds are established for "matching" these schedules. It is shown that the area of interconnection obtained using the classical routing techniques can be reduced often for the VLSI modules if suitable permutation networks are used. The second issue is concerned with the changes in tradeoffs when the locations of input and output ports are prespecified for a VLSI circuit. A class of functions is introduced and it is shown that if a chip computes a function belonging to this class, then it may have an increased area-time tradeoff under unfavorable I/O specifications. However, for most functions, a chip having prespecified I/O ports may differ only by a constant factor in the used resources from a chip without such specifications. The third topic addressed is the conversion of non-boundary layouts into boundary layouts. Resource efficient conversions are provided for this case.

This extended abstract has four sections. In section I, we describe the VLSI model briefly, review some basic definitions and discuss the problems regarding I/O placement in detail. In section II, we derive close bounds for the first problem. Section III introduces a class of functions and discusses the tradeoffs for this class for specified I/O placement. Section IV discusses the conversion of non-boundary layouts into boundary layouts , presents a discussion and offers some unresolved problems. Detailed proofs and related results will appear in the final draft of the paper.

I. The VLSI models and issues concerning I/O placement

The VLSI model

The underlying model used is a generalization of Thompson's model [Th79, Ag83]. The unit of information is either a word (as in sorting or merging) or a bit (as in cyclic shift). If the unit is a word, then a processor consists of a constant number of registers and a processing unit that performs simple arithmetic and boolean operations in unit time. Also, in this case, the function f computed by the chip is from $\{0, \ldots , N-1\}^n$ to $\{0 , \ldots , N-1 \}^m$ for some $N \geqslant n^{1+\epsilon}$ with $\epsilon > 0$. If the unit is a bit then each node consists of a finite state automaton (which, in turn consists of a constant number of flipflops and logic gates) and the function f computed by the chip, in this model, is a Boolean function i.e $f:\{0,1\}^n \rightarrow \{0,1\}^m$. If the unit is a bit (a word) then the model is called the bit model (the word model, respectively). Unless stated otherwise, we will be dealing with the word model only. The delay along every wire is taken to be unity. The area of the chip H is the minimum area of its bounding rectangle, the period P is the minimum time interval elapsed between the feeding of the first input bits of two consecutive data instances, and the worst case time T is the time elapsed between the input of the first input bit and the emergence of the last output bit for the most time consuming instance. If all the input/output ports lie on the periphery of a chip, then the chip is said to be a boundary layout. Also, if only one word is input to an input port, or output from an output port, then the port is said is to be dedicated.

Problems concerning I/O placement

The broad issue of I/O placement occurs frequently in the following situations .

[1] I/O Incompatibilities : In practice, the design of a large chip H is accomplished in three stages. In the first phase, known as the segmentation phase, H is segmented into smaller modules $H_1 \cdots ,H_k$ and each of these modules is designed separately. The second phase (or the placement phase) places these modules so that those modules that communicate with each other extensively are placed nearby. Finally, in the interconnection phase, all these modules are interconnected suitably to yield the chip H. The segmentation phase often causes the location and the schedule of inputs and outputs in these modules to be incompatible. This, in turn, may require adjustments and large interconnection area in the final phase. In view of the possible I/O incompatibilities, we identify bounds for the following problem :

Let two VLSI modules H' and H'' compute functions f' and f'' in areas and times A', T' and A'', T'' respectively. Furthermore, let H be constructed by cascading H' and H'' so that it computes the composition of the functions f'and f''. Then it is an important problem to estimate the increase in area and time, within a constant factor, for H as a result of the different input/output requirements of the individual modules.

[2] Specified I/O locations : A special purpose chip is usually designed from smaller, general purpose modules which are readily available in the integral form. These modules cannot be altered and have prespecified locations for input/output ports on the boundary. Consequently, the designer has to account for the given I/O locations when designing additional hardware for the special purpose chip. This may often lead to an increase in the area of the resulting chip as is evidenced from the example given below :

The intent is to design a special purpose chip that checks whether or not
$(c_0 . . . , c_{(n/2)-1}) = (h_0 . . . , h_{(n/2)-1})$ and $(c_{n/2} . . . , c_n) = (h_n . . . , h_{n/2})$
where $(c_n, \cdots ,c_0) = (a_{n-1}, \cdots ,a_0) + (b_{n-1}, \cdots ,b_0)$
$(h_n, \cdots ,h_0) = (d_{n-1}, \cdots d_0) + (e_{n-1}, \cdots ,e_0)$
and $a_i, b_i, d_i, e_i, c_j, h_j \epsilon \{0,1\}$; $0 \leqslant i < n-1$ and $0 \leqslant j \leqslant n$.

Assume that Brent and Kung's [BK82] VLSI adder circuit is provided. This circuit adds two n-bit numbers in $O(n \log n)$ area and in $O(\log n)$ time and has input and output ports on the boundary as shown in figure 1(a). Then there are two possible ways of designing the required chip :
(i) The chip is built directly from the specifications of f.
(ii)Two adder circuits for computing c_j's and h_j's are used and additional circuitry is built to check the equality of $c_{((n/2)+j)}$ and $h_{(n-j)}$ for $0 \leqslant j \leqslant (n/2)$ and of h_j and c_j for $0 \leqslant j < (n/2)$.

It is easy to see that the former approach for designing H is time consuming and requires an in-depth knowledge of constructing adder circuits but eventually leads to a a chip of $O(n \log n)$ area. On the other hand, using the given adder circuits saves some construction time and does not require the knowledge of adding circuits, but we will show later that this results in overall area of $O(n^2)$ for the desired chip. (See figure 1(b).) Keeping this in view, we study the following problem :

Let two boundary layouts H^* and H^{**} of areas A^* and A^{**} compute a function g $:\{0,...,N\text{-}1\}^n \longrightarrow \{0,...,N\text{-}1\}^m$ in times T^* and T^{**}, respectively. Furthermore, let H^* have specified I/O locations and schedule whereas H^{**} does not. Then two issues of interest are :

(i) The class of functions, g, for which the area-time tradeoffs for H^{**} differ from those for H^* by more than a constant factor.

(ii) The amount of increase in the tradeoffs for H^* with respect to those for H^{**}.

In the example of figure 1, g can be formulated as a Boolean function that checks the equality of c_j's and h_j's in the manner described above.

Note the similarity of these two problems related to I/O placement. The only difference is that in the former problem, two modules are cascaded and the designer has to match the I/O schedules, whereas in the latter problem, the I/O schedule and locations are prespecified for a chip and additional circuitry has to be suitably designed. Because of the essential similarities of these problems, the underlying techniques in establishing the lower bounds and in constructing the upper bounds are also similar, as will be seen in sections II and III. However, we treat them separately to reveal some important features.

[3] Boundary vs. non-boundary layouts : The present technology for fabricating VLSI chips requires the input/output ports to be placed on the periphery of a chip. On the other hand, many VLSI algorithms have been provided for chips which may be non-boundary layouts. Thus, we consider the following problem :

Let H_1 and H_2 be two chips of area A_1 and A_2 that compute an arbitrary function g $:\{0 ,..., N\text{-}1\}^n \longrightarrow \{0 ,..., N\text{-}1\}^m$ in times T_1 and T_2, respectively. Furthermore, let H_1 and H_2 be boundary and non-boundary layouts, respectively. Then estimate the increase in the area-time tradeoffs of H_1 with respect to those for H_2.

II. Bounds for Cascade networks with I/O incompatibility

Theorem 2.1 : Let the dimensions of H' and H'' be (l',b') and (l'',b'') respectively. Then a chip H, of dimensions (l,b), obtained from cascading H' and H'' satisfies the following relationships for some I/O schedules of H' and H'' :

(a) $T = \Omega(T'+T'')$.

(b) $l = \Omega(l'+l''+(n/T))$.

(c) $b = \Omega(b'+b''+(n/T)+(n/l))$.

Corollary 2.2 : If H' and H'' have a boundary layout and if a chip H is obtained by cascading H' and H'', then the three relationships stated for theorem 2.1 also hold for H.

Corollary 2.3 : If H' [or H''] has a boundary layout with each output word [input word,respectively] having a dedicated port, then the chip H obtained by cascading H' and H'' obeys $AT = \Omega(n^2)$ for some I/O schedule of H' and H''.

Theorem 2.4 : Let two boundary layouts H' and H'', of dimensions (l',b') and (l'',b'') compute the functions f' and f'' in times T' and T'', respectively. Then a chip H that is obtained by cascading H' and H''and is of length (l,b) and computes the composition of f' and f'' in time T, satisfies the following relationships for all I/O schedules of H' and H'':

(a) $T = O(T' + T'' + \log n)$.

(b) $l = O(l' + l'' + (n/T))$.

(c) $b = O(b' + b'' + (n/T) + (n/l))$.

Proof : We establish the above statement in three steps. In lemma 2.5, we show that if a boundary layout network N performs the Descend operation [PV81] in T time steps, then a network N', obtained by cascading three identical copies of N, performs any permutation in at most 3T time steps. In lemma 2.6, we present the modified Cube Connected Cycles as a network of dimensions $O(n/T)$ *by* $O(n/T)$ that performs Descend in T time steps for $\Omega(\log n) = T = O(\sqrt{n})$. It may be noted that the simple cube connected cycles [PV81] can also be used for this purpose. However, the modified CCC's have two advantages: (1) it is more regular than the CCC and (2) the modified CCC can be embedded so that each wire has $\Theta(n/T^2)$ length. For the ordinary CCC, the maximum edge length is $O(n/T)$ and lengths of various wires differ by more than a constant factor. Finally, in lemma 2.7, we modify the modules H' and H'' suitably and interconnect them using three copies of the modified CCC in cascade, thereby, making the I/O schedules of the two modules compatible.

Lemma 2.5 : If a network N performs the Descend operation in T time steps, then any permutation of n words can be performed in 3T time steps by suitably executing Descend, thrice, on three cascaded copies of N.

A $(k^2 : n/k^2)$ - modified cube connected cycles is realized as follows :
First (n/k^2) disjoint copies of a k by k systolic array are stacked so that for any i , j , k with $1 \leqslant i,j \leqslant k$ and $\Omega(\log n) = k = O(\sqrt{n})$, the (i,j)-th nodes of all these copies are in a straight line. Next, all the (i,j) nodes for a given i and j are connected so that they form a CCC of (n/k^2) nodes. Figure 2 shows a (4,8)-modified CCC schematically with the CCC connections being shown for all the nodes numbered (1,2) in the 8 copies of the 2 by 2 array.

It can be easily seen that a $(k^2 : n/k^2)$-modified Shuffle Exchange network can be constructed similarly by realizing a Shuffle Exchange network for each set of nodes with the same (i,j) indices. The degree of the nodes in the modified CCC is bounded by seven but this degree can be easily restricted to four by replacing every node in the network by a suitable triplet of vertices. The layout of a $(k^2 : n/k^2)$-modified CCC is obtained as follows :

Embed each CCC of (n/k^2) nodes in $16(n^2/k^4)$ area so that the embedded nodes lie along the diagonal of this region and each edge is $\Theta(n/k^2)$ long. Next, take a rectangular grid of $16(n^2/k^2)$ area, partition this in k^2 equally sized regions and place an embedded (n/k^2) CCC in each of these regions. Finally, add the connections belonging to the systolic arrays of the modified CCC by increasing the area by at most four times. Again, it is easy to see that even if the degree of each node in the CCC is bounded by four, this network can be still embedded with the length of each edge as $\Theta(n/k^2)$.

Lemma 2.6 : A $(k^2 : n/k^2)$-modified CCC performs the Descend operation in $O(k)$ steps for $\Omega(\log n) = k = O(\sqrt{n})$.

Definition: If a word output by H' is not used by H'' in its computation, then this word is called *redundant*. Otherwise, the word is called *non–redundant*.

Lemma 2.7 : Let a boundary layout H''' of dimensions (l''', b''') compute a function f in

T”’ time units. Then for any T_4''', l_4''', b_4''' such that $T_4''' \geqslant 4T'''$, $l_4''' \geqslant 16(l''+\sqrt{n})$, and $b_4''' = O[(n/T_4''')+(n/l_4''')+b_4''']$, there exists another layout H_4''' of dimensions (l_4''', b_4''') that computes f in T_4''' time units. Furthermore, each output port of H_4''' outputs $T_4'''-1$ words, the total number of such ports is $(2n/T_4''')$ and exactly n of the total output words are non-redundant.

Corollary 2.8 : Let H’ and H” be boundary layouts such that H’ has a constant aspect ratio and let these modules be cascaded to result in a chip H which computes the composition of the functions computed by H’ and H”. Then the area and time of H are different from those of H’ or H” by at most a constant factor.

Theorem 2.9 : Let H’ and H” of dimensions (l',b') and (l'',b'') respectively, be either boundary layouts or have dedicated ports. Then there exists a chip H which computes the composition of two functions computed by H’ and H” in T time units and is of dimensions (l,b) s.t. for any T with $T = \Omega(T'+T'')$, $l = \Omega(l'+l''+(n/T))$ there exists a b with $b = O(b'+b''+(n/T)+(n/l))$.

Theorem 2.10 : Let H’ and H” be two modules of area A’ and A” that compute the functions f’ and f” in T’ and T” time steps respectively. Then there exists a chip H of dimensions (l, b) which computes the composition of f’ and f” in T time steps s.t. for any
$T \geqslant 16(T' + T'' + logn)$,
$l = O(l' + l'' + [\sqrt{(A' + A'' + n)}\log(A' + A'' + n)] + (n/T))$
and $b = O(b'+b''+\sqrt{(A'+A''+n)}\log(A'+A''+n)+(n/T)+(n/l))$.

III. Bounds for specified I/O locations

Definition : A function $y = f(x_1, \cdots ,x_n)$ with $y \in \{0,1\}$ and $x_1, \cdots x_n \in \{0, \ldots N\text{-}1\}$, is said to be a k-pairwise related if there exists k disjoint pairs (x_1,x_2), ..., (x_{2k-1},x_{2k}) such that for each pair (x_{2i-1},x_{2i}) with $1 \leqslant i \leqslant k$, there exist at least N^{k-1} assignments to the remaining variables that cause y to be zero or one depending on whether or not x_{2i-1} equals x_{2i}, respectively. It is easy to see that checking the equality of $(x_1, \cdots ,x_{n/2})$ and $(x_{(n/2)+1}, \cdots ,x_n)$ and determining if any x_i equals x_j for $i \neq j$ can be redefined as (n/2)-pairwise related functions.

Theorems similar to those established in section II can also be stated. Also, the lower bounds given above can be easily extended for a multifunction $f{:}\{0,\ldots,N\text{-}1\}^n \longrightarrow \{0,\ldots,N\text{-}1\}^m$ by suitably modifying the definition of a k-pairwise related function. These lower bounds can be extended to hold for Boolean functions. However, demonstrating optimal bounds for these problems remains open in the bit model (i.e. for Boolean functions). It can be seen that any simple minded approach for converting the upper bound constructions in the word model to those in the bit model increases the $area-(time)^2$ bounds by a factor of $O(log^2 n)$, thereby leaving a gap of the same order in the two bounds. In the bit model, counting until T time steps and comparing these counts (at various stages of the computation) has to be performed. This results in each node requiring an $O(logn)$ area and the overall time being multiplied by a factor of $O(log^{1/2} n)$. Consequently, the $area-(time)^2$ increases by a factor of $O(log^2 n)$ in the bit model.

The problem of specified I/O ports was encountered by Kosaraju and the author [Ko81] when they were trying to design a chip H_m which multiplies two n-bit numbers in $\Theta(logn)$ time and occupies $O(n^2 \log^k n)$ area for some constant k. The intended approach involved the computation of n partial products - one for each of the multiplier bits - and then summing up these partial products using a (3,2) technique described in [Of63]. This, in turn required the design of a chip H_a which adds two n-bit numbers $(a_0, \cdots ,a_{n-1})$ and $(b_0, \cdots ,b_{n-1})$ such that the input and output bits have dedicated ports and if the periphery of the chip is traversed in the clockwise direction starting at the port corresponding to the bit a_0, the encountered ports correspond to the bits in the following sequence : $(a_0, \cdots, a_{n-1}, \ldots, b_0, \cdots, b_{n-1})$. Subsequently we established that, in the bit model, for any H_a (with the above described location of I/O ports), $AT = \Theta(n^2)$ holds for $\Omega(logn) \leqslant T \leqslant O(n)$.

Recently, we discovered that Savage [Sa81] had also observed *the change in area of the chip that adds two n-bit numbers and has prespecified input/output ports.* However, in view of the remarks given above, it is easy to see that the following statement due to him is incorrect.

[Sa81] "As a last observation, we show that if some natural constraints, such as contiguity of input and output variables on the periphery of the chip, are placed on VLSI algorithm, then the area required by them can be excessively large. This is demonstrated with the binary addition function with Boolean variables representing each of the three integers- the two inputs and the result- are required to be contiguous on the boundary of the chip. In this case it is easy to show that many node disjoint paths must exist between inputs and outputs and these must cross at many points. This leads to a lower bound on the chip area which is quadratic in length of the inputs."

IV. Discussion and Extensions

[a] Boundary and non-boundary layouts

To transform the non-boundary layouts into boundary layouts, theorems 2.9 and 2.10 can be used. These results are stated in theorems 4.1 and 4.2.

Theorem 4.1 : Let a non-boundary layout H_1 of dimensions (l_1, b_1) compute an arbitrary function f in time T_1. Furthermore, let H_1 have dedicated ports. Then, there exists a boundary layout H_2 that computes f in cT_1 time steps (for some constant c) and has $O(((n/T)+l_1)b_1)$ area.

Theorem 4.2 : Let a non-boundary layout H_1 of dimensions (l_1, b_1) compute an arbitrary function f in time T_1. Then there exists a boundary layout H_2 that computes f in cT_1 time steps (for some constant c) and has either $O(nlog(A_1+n)^* \sqrt{(A_1+n)}/T_1 + (A_1+n)log^2(A_1+n))$ area (if $A_1 \leqslant n^2$) or has A area.

Let $A=\Omega(n)$. Then it is easy to see that theorem 4.1 is optimal for $area-(time)^2$ product if this product is $\Omega(n^2)$ for H_1. Also, if $T_1 = \Omega(\sqrt{n})$, then the dimensions of H_2 are equal, within a constant factor, to those of H_1. For $\Omega(log^2 n) = T = O(\sqrt{n})$ we show that the results of theorem 4.1 are optimal within a factor of $O(b_1/\sqrt{n})$. Following Nath et al., we have :

Theorem 4.3 : An $(m/logm)$-Orthogonal tree cycle of $O(m^2)$ area computes the connected components of an m vertex graph in $O(log^3 m)$ time when the adjacency matrix of the graph is provided to the chip in a where and when oblivious manner.

The proof of this theorem can be found in [NMB81]. Also, from [Ja81] we quote, without proof, the following :

Theorem 4.4 : Let the adjacency matrix of an m-vertex graph be provided as input to a chip H in a where and when oblivious manner, and let this chip compute the connected components of the given graph. Then this chip satisfies $AT = O(m^2)$.

Theorem 4.5 : There exists a function f such that if a non-boundary layout chip, H_1, of area A_1 computes f in T_1 time steps then there does not exist any boundary layout H_2 which computes f in $o(n\sqrt{n}/T)$ and O(T) time steps (for $\Omega(\log^3 n) = T$) where n is the

---

Two functions f(n) and g(n) are related as f(n) = og(n) or as $\omega(f(n))$ if f(n) < cg(n) for any constant c.

size of the input fed to the chip.

[b] Remarks and open problems

I/O Incompatiblities : To match the I/O schedule and locations of two cascaded modules, the lower bound was established using the conventional, information flow technique. The upper bound, on the other hand, was identified by modifying these modules suitably and then connecting them through a pipelined permutation network. Regarding these bounds, the following points require mention :

(i) The stated bounds are optimal, within a constant factor, for many I/O schedules given in section II. However, when one of the modules is a non-boundary layout and its ports are not dedicated, there remains a gap of $O(\sqrt{n})$ in the two bounds, for the worst case.

(ii) Permutation networks have been used previously by Bhatt and Leiserson [BL82] in configurable layouts, by Leighton [Le82] in area-efficient layouts and by Rosenberg [Ro81] in reconfigurable and fault-tolerant networks. However, conventional permutation networks like the Benes and the Omega networks [Wa68,La76] occupy $\Theta(n^2)$ area. Our contribution was to pipeline the permuter suitably so that it occupies much less area, thereby reducing the area of interconnection obtained using the classical routing techniques for VLSI modules.

(iii) The modified cube connected cycles and the modified Shuffle Exchange network demonstrate the existence of networks that perform an arbitrary permutation in $\Theta(\log n)$ time and have wires of $\Theta(n/(\log^2 n))$ length. Also, within a constant factor, all embedded wires of these networks have the same length, and this makes the networks useful in synchronous non-pipelined circuits [Ag83a]. To embed the Cube Connected Cycles and the Shuffle Exchange networks such that their wires are $\Theta(n/(\log^2 n))$ long remains unresolved.

(iv) While identifying bounds for the cascade networks, we have assumed that there is a one-to-one correspondence between the output words of one module and the input words of the other. The extended case of a many-to-one correspondence (i.e. when an output of the first module is input at several input ports and/or at several time instants of the second module) also requires thorough investigation because of its vast applicability. The latter problem is analogous to the multiterminal net routing in the same sense as the former is analogous to the single-terminal routing problem.

I/O Specifications : For this problem, we showed that if a function f of n variables is an $O(n)$-pairwise related function and if the two boundary chips H^* and H^{**} compute the function f so that the I/O ports are specified for H^* but not for H^{**}, then their $area-(time)^2$ product may differ by a factor of $O(n/h(n))$, for some function h(n) of $O(n)$ and for some specifications of I/O ports. We also showed the optimality of the above bound by demonstrating that there always exists the required chip H^* such that $A_1^* T_1^{*2} = O(A_2^{**} T_2^{**2} + n^2)$. However, the following points are worth mentioning :

(i) There exists a gap in the $area-(time)^2$ products of H_1 and H_2 when the function f (of n variables) is k-pairwise related. Also, a large gap, similar to that for the I/O incompatibility problem remains, when H^* and H^{**} are non-boundary layouts.

(ii) No bounds are known when multiple copies of inputs are given to H^* and H^{**} and the ports of the two chips are prespecified. That multiple copies reduce the $area-(time)^2$ for transitive functions [Vu80] has been shown by [KZ81] and [Ke82].

(iii) It is quite likely that the upper bound constructions for all three problems regarding I/O placement may not be practically useful. However, theoretically, these bounds indicate that if any function has an $area-(time)^2$ complexity of $\Omega(n^2)$, then this complexity remains unchanged apart from a constant factor, inspite of adverse I/O specifications. Unfortunately, we know of no functions which have an $area-(time)^2$ complexity of $\omega(n^2)$.

(iv) Finally, for all the three topics regarding I/O placement, it would be useful to establish area-period-time tradeoffs.

Acknowledgements

The author is grateful to Prof. S.R.Kosaraju for funding this research from his NSF grant No. MCS-820-5167 and for his valuable suggestions. He also thanks Prof. Joseph O'Rourke and Dr. Kathleen Callaway for reviewing this manuscript.

References

[Ag83a] A. Aggarwal, "Period-Time Tradeoffs for VLSI models with delay," to be presented at the 24th Annual IEEE Symposium on the Foundations of Computer Science, 1983.

[Ag83b] A. Aggarwal, "On Simulation and Interrelations between Boolean and VLSI circuits," Tech. Report in preparation.

[BL82] S. N. Bhatt and C. E. Leiserson, "How to assemble tree machines ," Proceedings of the 14th Annual Symposium on the Theory of Computing, May 1982.

[BK81] R. P. Brent and H. T. Kung, "The area-time complexity of binary multiplication," JACM, 28, 3, July 1981, pp. 521-534.

[BK82] R. P. Brent and H. T. Kung, "A regular layout for parallel adders," IEEE Transactions on Computers, March 1982, pp. 260-264.

[Ja82] J. Ja'ja', "The VLSI Complexity of graph problems," Tech. Report No.CS-81-25, Oct. 81, Department of Computer Science, Penn State University.

[Ke82] Z. Kedem, "Optimal Allocation of Computational Resources in VLSI," Proceedings of the 23rd IEEE Annual Conference on Foundations of Computer Science, Nov. 1982.

[Ko81] S. R. Kosaraju, Personal Communication.

[KZ81] Z. M. Kedem and A. Zorat,"On relations between input and communication computation in VLSI," Proceedings of the 22nd Annual Symposium on the Foundations of Computer science, Oct. 1981. pp. 37-44.

[La75] D. H. Lawrie, "Access and alignment in an array processor," IEEE Transactions on Computers, Vol. 24, pp. 1145-1155, Dec. 1975.

[Le82] F. T. Leighton, "A Layout Strategy for VLSI which is Provably Good," Proceedings of the 23rd IEEE Annual Symp. on the Foundations of Computer Science, Nov. 1982.

[Le80] C. E. Leiserson, "Area Efficient Graphs Layouts (for VLSI)," Proceedings of the 21st Annual IEEE Symposium on the Foundations of Computer Science, Oct. 1980, pp. 270-281.

[LS81] R. J. Lipton and R. Sedgewick, "Lower bounds for VLSI," Proceedings of 13th Annual Symposium on the Theory of Computing, May 1981.

[NMB81] D. Nath, S. N. Maheshwari and P. C. P. Bhatt, "Efficient VLSI Networks for Parallel Processing based on Orthogonal Trees," unpublished manuscript.

[Of63] Yu. Ofman, "On the Algorithmic Complexity of Discrete Functions," Soviet Physics - Doklady, Vol. 7, No. 7, Jan. 1963.

[Pa80] D. S. Parker, "Notes on Shuffle/Exchange type Switching Networks," IEEE Transactions on Computers, vol. 29, No. 3, pp. 213-222, March 1980.

[PV81] F. P. Preparata and J. E. Vuillemin, "The cube-connected cycles, a versatile network for parallel computation," Proceedings of 20th Annual Symposium of the Foundations of Computer Science, Oct. 1979, pp. 140-147.

[Ro81] A. Rosenberg, "Routing with permuters : toward reconfigurable and fault tolerant networks ," Tech. Report, CS-1981-13, Duke University, 1981.

[Sa80] J. E. Savage, "Area-time tradeoffs for matrix multiplication and related problems in VLSI models," Tech. Report CS-50, Department of Computer Science, Brown University, Aug. 1979.

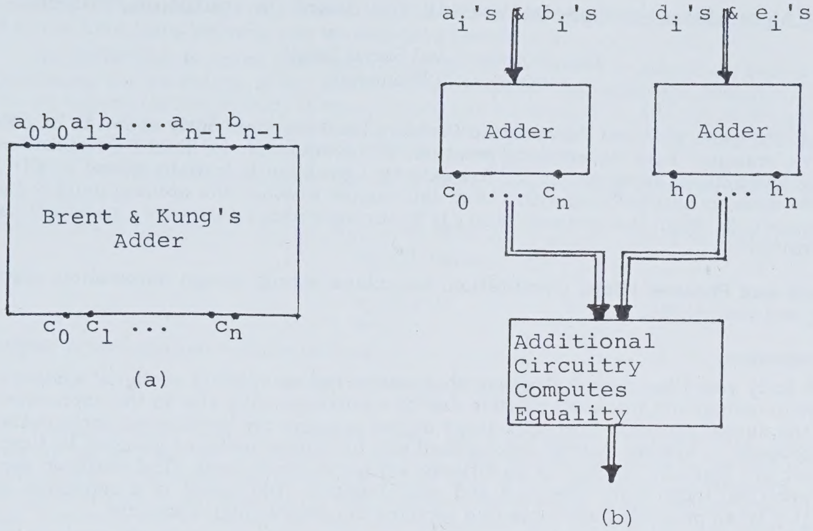
[Sa81] J. E. Savage, "Planar Circuit complexity and the performance of VLSI algorithms," in VLSI Systems and Computations, edited by H. T. Kung, B. Sproull, and G. Steele, Computer Science Press, Oct. 1981, pp. 61-68.

[Th79] C. D. Thompson, "Area-time complexity for VLSI," Proceedings of the 11th Symposium on the Theory of Computing, May 1981, pp. 81-88.

[Th80] C. D. Thompson, "A Complexity theory for VLSI," PhD dissertation, department of Computer Science, Carnegie Mellon University, 1980.

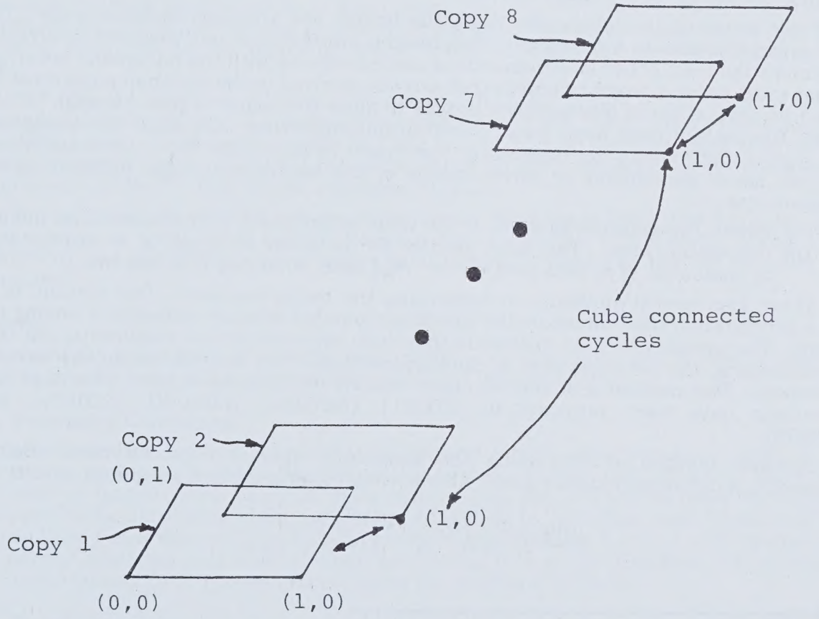
[Vu80] J. E. Vuillimin, "A Combinatorial limit to the computing power of VLSI circuits," Proceedings of the 21st Annual IEEE Symposium on the Foundations of Computer Science, Nov. 1980, pp. 294-300.

[Wa68] A. Waksman, "A Permutation network," JACM 15, 1, July 1968, pp. 159-163.



a0b0a1b1...an-1bn-1
Brent & Kung's Adder
c0 c1 ... cn
(a)
ai's & bi's
di's & ei's
Adder
Adder
c0 ... cn
h0 ... hn
Additional Circuitry Computes Equality
(b)

Fig. 1: (a) Brent & Kung's adder requires interleaving of bits.
(b) The additional circuitry to compute the equality of $(c_0,\dots c_{n/2}, c_{n/2+1}\dots c_n)$ and $(h_0,\dots h_{(\frac{n}{2}-1)}, h_n,\dots h_{n/2})$ requires $\Omega(n^2)$ area.



Copy 8
Copy 7
(1,0)
(1,0)
Cube connected cycles
Copy 2
(0,1)
Copy 1
(1,0)
(0,0)
(1,0)

Fig.2: A (4,8)-Modified Cube Connected Cycles. Nodes numbered (1,0), form a CCC on 8 nodes for every copy. Similarly, (0,0), (0,1) & (1,1) also form a CCC on 8 nodes.

Exact Algorithms For Special Cases Of The Board Permutation Problem*

James Cohoon+ and Sartaj Sahni
University of Minnesota

Abstract: The general Board Permutation Problem has been previously shown to be NP-hard. We consider here determining whether an instance of the Board Permutation Problem has optimal density 1 or 2. The density 1 problem is trivially solved in $O(n)$ time. We develop an $O(n^3)$ algorithm that determines whether the optimal density for an instance is 2. When the optimal density is 2, our algorithm constructs a permutation with density 2.

Keywords and Phrases: board permutation; backplane wiring; design automation; algorithms; and complexity.

1. Introduction

The last forty years has seen a dramatic increase in the complexity of digital systems. This rise in complexity has been possible due to a corresponding rise in the sophistication of the design process. Currently, large digital systems are partitioned early in the design process. A system may be decomposed into functional units, or grouped by functions that are logically similar, or partitioned on some other basis. The partition elements are then individually designed and implemented. The result is a collection of *boards* that when properly interconnected serve as the desired digital system.

Various subsets of the boards may require that information be shared for the operation of the overall digital system. Each such subset is called a *net*. The information is called a *signal*.

The boards in a net are interconnected by wires. A *pin* is an interconnection point on a board. As several boards may share several signals, it is possible there are several connections from one board to another where each interconnection has its own pin. The pins in a net are all *electrically common*.

Prior to wiring the interconnections, the boards are arranged in some order. The arrangement results in a *backplane*. Besides the interconnections required to directly implement the nets, other interconnections are introduced with the backplane ordering. If a net must cross a board or boards that are not involved in the net then a pin must be placed on each of these intermediary boards to allow the signal to pass through. If the boards themselves have been constructed to minimize their size then the backplane area may be minimized by minimizing the space required for the interconnections. Different linear placements or *permutations* of the boards may have different space requirements.

The *Board Permutation problem* or *BP* is to determine a permutation that minimizes the backplane area. The input to the BP problem is a set of n boards $B = \{b_1 \cdots b_n\}$ and a set of m nets $L = \{N_1 \cdots N_m\}$ on B, where $N_i \subseteq B$, $1 \le i \le m$.

There are several methods for minimizing the backplane area. One method is to find a permutation that minimizes the maximum number of interconnections among the boards. The premise for this method is that each interconnection requires space, and by minimizing the total number of interconnections, the overall backplane area is minimized. This method and several other related methods have been examined and algorithms have been proposed in [STEI61], [RUTM64], [ADOL72], [CEDE74], and [SANG75].

Another method of minimizing the backplane area is explored here. Before proceeding a definition must be given. The *density*(π), where π is a permutation of B, is:

$$\max_{1 \le i \le n-1} \left\{ \left| \left(\bigcup_{j=1}^{i} S_{\pi_j} \right) \cap \left(\bigcup_{j=i+1}^{n} S_{\pi_j} \right) \right| \right\}.$$

---

* This research was supported in part by NSF grant MCS 80-005856.

+ Present address: Dept of Applied Mathematics and Computer Science, University of Virginia, Charlottesville.

where $S_i = \{ \alpha \mid \alpha \in L, b_i \in \alpha\}$, $1 \le i \le n$. Informally, the *density(π)* is the maximum number of interconnections between any two adjacent boards in π.

In [CEDE74] it is shown that minimizing the interconnection density is equivalent to minimizing the backplane area. Therefore, to minimize the backplane area it suffices to choose a permutation π from Π whose density is minimal (Π is the set of all permutations of B). This method of minimizing the backplane area has been previously examined in [GOTO77] and it is the subject of further exploration here.

With the terms defined above the BP problem may be expressed formally in the following manner:

Input: A set of boards $B = \{b_1 \cdots b_n\}$; a set of nets $L = \{N_1 \cdots N_m\}$ on B such that:

$$N_i \subseteq B, \ 1 \le i \le m$$

$$\bigcup_{i=1}^{n} N_i = B$$

Output: A permutation π of B such that:

$$density(\pi) = \min_{\sigma \in \Pi} \left[\text{density}(\sigma) \right]$$

The algorithms that are developed here and elsewhere to minimize the backplane area by minimizing the maximum density of interconnection, have a variety of other applications. One recent application is in the *Gate Array Layout* or *GAL problem*. In the GAL problem the input is a set of cells, where a *cell* is some basic electrical and logical unit. The cells are to be laid out and interconnected in a regular fashion, typically in a matrix-like form. This structure, a *gate array*, may be built both economically and with fast turnaround [TODD81]. The gate array has both interconnections among the columns of cells and intraconnections within a column of cells. A part of the output solution is a gate array whose size is minimal. Algorithms for the BP problem may be adapted to determine both the order of columns and the order of cells within a column for a minimal sized gate array.

The BP problem is a generalization of the *Optimal Linear Arrangement problem* or *OLA*. OLA restricts the size of the N_i's in L, such that $|N_i| = 2$, $1 \le i \le n$. Such a restriction reduces the input to the form of an undirected graph. Stockmeyer has shown OLA to be NP-hard [GARE79]. As any algorithm for BP is also an algorithm for OLA, BP is also NP-hard.

For the OLA problem, [GURA81] contains a dynamic programming algorithm that determines whether there is a permutation with density k in $O(n^k)$ time. This algorithm also finds a permutation with density k if one exists. It is an open question whether the algorithm presented in [GURA81] can be modified to produce solutions for the general BP problem in $O(n^{ck})$ time, where c is some constant.

In this paper, we develop two polynomial time algorithms Bpd1 and Bpd2 that can, respectively, be used when the BP instance has a solution with density 1 and 2. In [COHO82b] and [GOTO77] several approximations algorithms for the BP problem are presented.

2. Bpd1

Bpd1 assumes it has been previously verified that L is nonempty. If L is empty then the optimal density is 0 and any permutation achieves this density.

2.1. Necessary Conditions

For a permutation to have a density of 1, several significant restrictions exist on the form of the problem instance. One restriction is that no board may appear in more than two nets, as from a given board at most one net can exit left and at most one net can exit right if the density is to be 1. Another restriction is that there must be at least two boards i and j in B where $|S_i| = |S_j| = 1$, as the boards that are leftmost and rightmost in the permutation can only have a single net exiting in a single direction. These checks can be performed in *O(n)* time and compose the first part of *Bpd1*.

2.2. Board Groupings

Bpd1 groups the boards in several ways. The boards within a group are processed together. Several definitions appear below. These definitions will help characterize the way the boards are grouped.

Definition: A board u is *part of* some net i iff $u \in N_i$.

Definition: A *component* C is a maximal subset of B such that for any two boards a and b in C, either a and b are part of the same net or there is a sequence of boards $b_{i_1}, \cdots b_{i_k}$, $1 \leq k \leq n$, such that a and b_{i_1} are part of some net in L; b and b_{i_k} are part of some net in L; and b_{i_j} and $b_{i_{j+1}}$ are part of some net in L, $1 \leq j \leq k-1$.

The component C that a board b appears in can be constructed in $O(|C|)$ time. Initially C is set to b. For each net i in S_b all boards in N_i are added to C. This addition to C is then continued in a similar manner by adding to C all boards u such that there is a board v in C where u and v are part of the same net. This addition is continued until no new boards can be introduced to C by the above rules.

Definition: A *chain element* is an ordering $(b_{i_1} \cdots b_{i_k})$ for some net a $(N_a = \{b_{i_1} \cdots b_{i_k}\})$ such that $S_{b_{i_j}} = \{a\}$, $2 \leq j <= k-1$, and $|S_{b_{i_k}}| \leq 2$.

Definition: A chain element $(b_{i_1} \cdots b_{i_k})$ is an *interior* chain element if $S_{b_{i_1}} = S_{b_{i_k}} = 2$.

Definition: A chain element $(b_{i_1} \cdots b_{i_k})$ is a *terminating* chain element if $S_{b_{i_1}} = 2$ and $S_{b_{i_k}} = 1$.

Definition: A chain element $(b_{i_1} \cdots b_{i_k})$ is a *singleton* chain element if $S_{b_{i_j}} = 1$, $2 \leq j \leq k$.

Definition: A chain element $(b_{i_1} \cdots b_{i_k})$ for net s *connects* to a chain element $(b_{j_1} \cdots b_{j_l})$ for net t if $b_{i_k} = b_{j_1}$.

Definition: A *chain* is either a singleton chain element or a maximal sequence $e_1, \cdots e_k$ of chain elements such that e_i and e_{i+1} are connected, $1 \leq i \leq k-1$; e_i is an interior chain element, $2 \leq i \leq k-1$; and e_k is a terminating chain element.

2.3. Processing A Component

Components may be processed in any order. When processing an individual component, we assign all its boards a contiguous segment of the permutation. A density 1 ordering for each component may be trivially obtained (if it exists) in time linear in the component size.

3. Bpd2

Bpd2 assumes the instance is a single component. The algorithm also assumes it has been previously verified by *Bpd1* that there is no permutation with density 1 for the component. If the desired instance is not a single component then *Bpd2* can be called repeatedly to process each component. If all components have density 2 or less then by laying the components out in the order they were processed, the overall permutation will have density 2.

3.1. Necessary Conditions

The checks that compose the first part of *Bpd2* are a verification that conditions (i) and (ii) of Lemma 3.1 are true for the problem instance.

Lemma 3.1 If (B, L) is an instance of the board permutation problem such that the optimal density for the instance is 2 then the following conditions hold:

i. $|S_i| \leq 4$, for all $i \in B$,

ii. $|S_i \cap S_j| \leq 2$, for all $i, j \in B$, $i \neq j$.

Proof: Obvious. ▪

The verification that condition (i) holds for an instance can be performed in $O(n)$ time. The verification that condition (ii) holds for an instance can be performed in $O(n^2)$ time. Therefore the first part of *Bpd2* operates in $O(n^2)$ time. Note that a consequence of condition (i) is that $m \leq 2n-2$. Therefore m is $O(n)$.

3.2. Processing Boards That Appear In A Single Net

Let $Q = B$. A board s is *active* if s is in Q. A board s is *inactive* if it is not in Q. Initially all boards are active.

Similarly, let $R = S$, where $S = \{S_1 \cdots S_n\}$. A net i is *active* if there exists a j in Q such that $i \in S_j$. If no such j exists then i is *inactive*. Initially all nets are active.

A net i is *externally active* if there exist boards j and k in Q, such that $j \neq k$, $i \in S_j$, and $i \in S_k$. If no such j and k exist then i is *externally inactive*.

Let $T_i = \{ b \mid b \in B,\ S_b = \{i\} \}$, $1 \le i \le m$. The set T_i is the set of boards that are only part of net i, $1 \le i \le m$.

A board b in Q is of ***type k***, if $|S_b|$ is k and no restrictions have yet been made on how the k nets exit b. Initially, all boards are of type 1, 2, 3, or 4. A type 5 board is introduced later, it will have restrictions on how its nets exit.

A type 1 board that appears in net a is represented by $\begin{bmatrix} a \end{bmatrix}$. A type 2 board that appears in nets a and b is represented by $\begin{bmatrix} a \\ b \end{bmatrix}$. A type 3 board that appears in nets a, b, and c is represented by $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$. A type 4 board that appears in nets a, b, c, and d is represented by $\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$.

The processing of boards that appear in only one net results in the deletion of all type 1 boards from the instance I to be solved. The resulting reduced instance is I'. In solving I', if for some net i, $i \in L$, $N_i - T_i = \{b\}$, for some $b \in Q$, then one track is reserved for net i to one side of b. The side is determined by *Bpd2* later. $N_i - T_i$ cannot be empty, as that would imply I can be wired with one track and this cannot be true as *Bpd1* would have found a permutation with density 1.

Given a permutation π' of density 2 for I', a permutation π of density 2 for I can be constructed as follows: if net i exits board s from the left then the boards in T_i may be inserted in any order immediately prior to s. If net i exits right then the boards may be placed immediately after s. Otherwise, net i only appears in the boards of T_i and s and by the convention from above one track on one side of s must be available, hence the boards of T_i may be placed on the side the free track occurs immediately adjacent to s. Clearly the new permutation's density will be 2. *Bpd2* uses the boolean variable h_{si} to indicate whether the boards in T_i are to be placed adjacent to s. If h_{si} is true and it has been determined the reduced instance has density 2 then the boards in T_i are placed by *Bpd2* in the manner described above at the conclusion of its fourth part.

The complexity is readily seen to be $O(n+m)$ which is $O(n)$.

3.3. Combining Boards That Share Two Nets

During the third part of *Bpd2* there are no type 1 boards to consider. The remaining boards may combine to produce type 5 boards. A type 5 board is shown in Figure 1. The letters a, b, c, and d in Figure 1 represent nets. Nets a and b exit one side of the board and the nets c and d exit the other side. There is no implication that a and b exit left and that c and d exit right. Note, it may be the case that $\{a,b\} \cap \{c,d\} \neq \phi$.

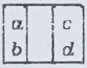


Figure 1 - Representation Of A Type 5 Board

In all combinations two boards say s and t are taken to produce a new board st, where st is a new symbol. The boards s and t are made inactive and the new board st is made active. A net i is made inactive if it appeared in the representation of s or t but is not part of st's representation. If a net is made inactive but is also part of some other active board's representation then it will be the case that no permutation with density 2 is possible. At the conclusion of the fourth part of *Bpd2* transformations are performed to produce a solution for the original problem instance if it has been determined the reduced instance has density 2.

The third part of *Bpd2* either results in the conclusion that no permutation with density 2 is possible or yields a BP instance $I'' = (Q,R)$ with the property that for any two boards s and t in Q, $|S_s \cap S_t| \le 1$. The combinations are performed in a manner that assures there is a permutation with density 2 in the reduced instance I'' iff there is a permutation with density 2 in the original instance I'. In addition a permutation with density 2 in the reduced instance can be modified in at most $O(n)$ time to produce a solution for the original problem.

Note that the combinations performed in this part introduce new boards and remove others. Therefore, the N_i's must be updated to reflect the change in boards.

How *Bpd2* combines two boards s and t is described below. In choosing a pair of boards to combine, preference is given to a combination that will not make any active nets inactive. Upon finishing processing one s and t combination the process of combining is iterated with another pair of boards. The iteration continues until no two boards in Q share more than one net or that it has been determined there is no permutation with density 2.

Note that $|S_s \cap S_t| = 2$ in the first ten cases we consider.

Case 1. $s = \begin{bmatrix}a\\b\end{bmatrix}$ and $t = \begin{bmatrix}a\\b\end{bmatrix}$

The result is $st = \begin{array}{|c|c|}\hline a & a\\ b & b\\\hline\end{array}$. As s and t share two nets the two tracks available between s and t are both in use. One track is devoted to net a, the other to net b. The only active boards that can come between s and t in a permutation with density 2 are some type 2 boards equal to $\begin{bmatrix}a\\b\end{bmatrix}$ or type 5 boards equal to $\begin{array}{|c|c|}\hline a & a\\ b & b\\\hline\end{array}$. But these boards can equally well precede or follow st. They all must be adjacent to each other in a permutation with density 2, but the order among them is immaterial. Preference is given to the combining of these boards. If it has been determined two nets a and b must exit from the same side of some board then combinations that allow these two nets to continue exiting in parallel together are given preference over other combinations.

Case 2. $s = \begin{bmatrix}a\\b\end{bmatrix}$ and $t = \begin{bmatrix}a\\b\\c\end{bmatrix}$

The result is $st = \begin{array}{|c|c|}\hline a & c\\ b & d\\\hline\end{array}$ where $d \in \{a,b\}$. To determine d's value, three sets A, B, and C are constructed. The sets are built iteratively in a manner similar to building a component. Initially C is set to $N_c - \{st\}$. For each b in C *Bpd2* then adds to C all active boards u, $u \neq st$, such that u is part of some net in S_b. This step is repeated until no more boards can be added to C. C is called the *transitive closure* of $N_c - \{st\}$. The transitive closures A of $N_a - \{st\}$ and B of $N_b - \{st\}$ are calculated in a similar manner. Note, the preference given to the combinations of cases 1, 3, and 4 ensures there is no type 2 board i equal to $\begin{bmatrix}a\\b\end{bmatrix}$ nor type 5 board j equal to $\begin{array}{|c|c|}\hline a & a\\ b & b\\\hline\end{array}$ in either A or B.

One of several subcases may apply. The details of the many subcases that arise can be found in [COHO82a].

Case 3. $s = \begin{bmatrix}a\\b\end{bmatrix}$ and $t = \begin{bmatrix}a\\b\\c\\d\end{bmatrix}$

The result is $st = \begin{array}{|c|c|}\hline a & c\\ b & d\\\hline\end{array}$. As in case 1, s and t share the two nets a and b. In a manner similar to case 1 the only active boards that can come between s and t in a permutation with density 2 are ones whose active nets are a and b. But, these boards can also be placed to one side of s.

Case 4. $s = \begin{bmatrix}a\\b\end{bmatrix}$ and $t = \begin{array}{|c|c|}\hline a & c\\ b & d\\\hline\end{array}$

The result is $st = \begin{array}{|c|c|}\hline a & c\\ b & d\\\hline\end{array}$. The validity of the combination follows from case 3.

Case 5 $s = \begin{bmatrix}a\\b\\c\end{bmatrix}$ and $t = \begin{bmatrix}a\\b\\d\end{bmatrix}$

The result is $st = \begin{array}{|c|c|}\hline e & f\\ c & d\\\hline\end{array}$. The preference given to the combinations of cases 1, 3, and 4 ensures there are no active $\begin{bmatrix}a\\b\end{bmatrix}$ or $\begin{array}{|c|c|}\hline a & g\\ b & h\\\hline\end{array}$ boards that require both an a and b connection. The case is divided into several subcases. These subcases assume that the transitive closure A of $N_a - \{st\}$ and the transitive closure B of $N_b - \{st\}$ have a null intersection. If $A \cap B$ is not empty then it must be the case that there is no permutation with density 2. The details of the subcases may be found in [COHO82a].

Case 6. $s = \begin{bmatrix}a\\b\\c\end{bmatrix}$ and $t = \begin{bmatrix}a\\b\\e\\f\end{bmatrix}$

The result is $st = \begin{array}{|c|c|}\hline c & e\\ d & f\\\hline\end{array}$. If a appears in some active board u's representation then d is set to a. If b appears in some active board v's representation then d is set to b. If both u and v exist then no permutation with density 2 is possible, *Bpd2* indicates this and halts. That no permutation with density could exist follows as st can only have one of nets a or b exiting from it, but for both of these nets there are active boards that must be connected to the net. If neither u nor v exists then d is set arbitrarily to either a or b, as neither net needs the pin.

As this combination makes some active net inactive, the combination is not performed while other combinations exist that do not deactivate nets.

Case 7. $s = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$ and $t = \boxed{\begin{smallmatrix} a \\ b \end{smallmatrix}}\boxed{\begin{smallmatrix} e \\ f \end{smallmatrix}}$

The result is $st = \boxed{\begin{smallmatrix} c \\ d \end{smallmatrix}}\boxed{\begin{smallmatrix} e \\ f \end{smallmatrix}}$. The validity of the combination follows from case 6. The selection of d is performed in a similar manner.

Case 8. $s = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$ and $t = \begin{bmatrix} c \\ d \\ e \\ f \end{bmatrix}$

The result is $st = \boxed{\begin{smallmatrix} a \\ b \end{smallmatrix}}\boxed{\begin{smallmatrix} e \\ f \end{smallmatrix}}$ The validity of the combination follows from case 3. Nets c and d are both made inactive. Therefore, this combination is not performed while there exist combinations that do make nets inactive.

Case 9. $s = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$ and $t = \boxed{\begin{smallmatrix} c \\ d \end{smallmatrix}}\boxed{\begin{smallmatrix} e \\ f \end{smallmatrix}}$

The result is $st = \boxed{\begin{smallmatrix} a \\ b \end{smallmatrix}}\boxed{\begin{smallmatrix} e \\ f \end{smallmatrix}}$. The validity of the combination follows from case 3. If c is not an element of $\{e,f\}$ then c is made inactive. Similarly if d is not element of $\{e,f\}$ then d is made inactive. If it is determined that either c or d is made inactive then this combination is not performed while there exists combinations that do not make nets inactive.

Case 10. $s = \boxed{\begin{smallmatrix} a \\ b \end{smallmatrix}}\boxed{\begin{smallmatrix} c \\ d \end{smallmatrix}}$ and $t = \boxed{\begin{smallmatrix} c \\ d \end{smallmatrix}}\boxed{\begin{smallmatrix} e \\ f \end{smallmatrix}}$

The result is $st = \boxed{\begin{smallmatrix} a \\ b \end{smallmatrix}}\boxed{\begin{smallmatrix} e \\ f \end{smallmatrix}}$. The validity of the combination follows from case 3. Also, if c is not an element of $\{a, b, e, f\}$ then it is made inactive. Similarly, if d is not an element of $\{a, b, e, f\}$ then it is made inactive. If it is determined that c or d is made inactive then this combination is not performed while there exists combinations that do not make active nets inactive.

All other cases that *Bpd2* considers imply that there is no permutation with density 2. With the combining of boards it is possible that two boards share more than two nets. If this is the case, no permutation with density 2 is possible. The remaining cases where the intersection is two all involve a type 5 board $\boxed{\begin{smallmatrix} a \\ b \end{smallmatrix}}\boxed{\begin{smallmatrix} c \\ d \end{smallmatrix}}$. One of the nets shared is from $\{a,b\}$. The other net is from $\{c,d\}$. Clearly then any permutation for the instance has a density greater than 2.

3.4. Building The Permutation With Density 2

The last part of *Bpd2* takes the elements of Q and either builds a permutation with density 2 or determines there is no permutation with density 2.

Let C be the set of active boards with at most two externally active nets. The boards in C are the only possible candidates to head a permutation with density 2. If a candidate r can head a permutation with density 2 up to two externally inactive nets can be assigned to the left of r. From the right of the candidate its other nets can exit.

A description of how *Bpd2* is to process a candidate r is given below. If it is determined that r cannot head a permutation with density 2 a new r is selected and examined. This selection and examination is repeated until either the list C is exhausted or a permutation with density 2 has been found.

If the list C is exhausted then *Bpd2* indicates there is no permutation with density 2 and halts. If a permutation with density 2 is found then the permutation is transformed to produce a solution for the original instance.

3.4.1. Processing A Candidate To Head A Permutation

To process r, *Bpd2* first constructs two sets Q' and R'. Q' equals Q and R' equals R. Q' and R' serve the role of Q and R for given r, as the information in Q and R must not be lost if r does not head a permutation with density 2. In this part of the algorithm a board or net is considered active if it is a member of Q' or R' respectively. When a board has been assigned its position within the permutation it is removed from R'.

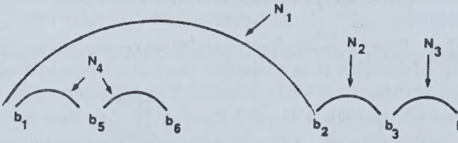
After constructing the two sets *Bpd2* attempts to build a permutation with r as the first board in the permutation. From r there are at most two externally active nets. With r and these net(s) *Bpd2* attempts to build chain(s) of type 2 boards. If these chains are built then a permutation with density 2 may be built by placing one chain of boards immediately next to r and the second chain immediately following the first. In Figure 2 there is an instance and permutation demonstrating this principle. However, in building the two chains we may reach a point where one or more of the conditions for a chain is violated. The two partial chains are still placed one after the other, immediately next to r. Suppose the nets at the end of the partial chains are a and b.

There are several ways for the violations to occur. Below is a description of how *Bpd2* handles these cases. The result of these cases is either a determination that there is no permutation with density 2 or a selection of which board s should occur next in the permutation. From s there will be at most two externally active nets exiting right. The chain building and case determination is then continued with these nets. The process is repeated until either a permutation with density 2 has been found or it is determined that r cannot head a permutation with density 2.

Case 1. a and b are externally inactive

$$
\begin{aligned}
B &= \{b_1, b_2, b_3, b_4, b_5, b_6\} \\
L &= \{N_1, N_2, N_3, N_4\} \\
n_1 &= \{b_1, b_2\} \\
n_2 &= \{b_2, b_3\} \\
N_3 &= \{b_3, b_4\} \\
N_4 &= \{b_1, b_5, b_6\}
\end{aligned}
$$

(a) - BP Instance



(b) Permutation For Above Instance With b_1 Heading Two Chains

Figure 2

As a and b are both inactive it must be the case that all boards have been assigned and a permutation with density 2 has been found. *Bpd2* then enters the next phase of this part and transforms the permutation into a permutation for the original instance.

Case 2. a and b are externally active and there exists s, $s \in Q'$, such that $\{a,b\} \in S_s$

There cannot be more than one board that appears in nets a and b since the previous part of *Bpd2* ensures $|S_i \cap S_j| \leq 1$, $i \neq j$. Board s occurs next in the permutation. There are several subcases depending on s's type. The details of these appear in [COHO83a].

Case 3. a and b are externally active and there is no s, $s \in Q'$ such that $\{a,b\} \in S_s$

All boards in N_a and N_b are of some type greater than 2. Therefore, the density between then next board in N_b and the next board in N_a must be at least 3. Thus a new r must be examined.

Case 4. b is externally inactive and a is externally active and there is a single board s for the next board that requires net a

There are several subcases depending upon board s's type.

Case 4.1 *s* is type 3

If *s* is of the form $\begin{bmatrix} a \\ b' \\ c' \end{bmatrix}$ then chain building and subsequent case determination is then performed with *b* ' and *c*'. As there was a single choice *s*, it is the case that *a* is now inactive.

Case 4.2 *s* is type 4

If s is of the form $\begin{bmatrix} a \\ b' \\ c' \\ d' \end{bmatrix}$ then one of *b* ', *c* ', or *d* ' must either be externally inactive or head a chain of type 2 boards that dies out. If not, a new *r* must be examined. Otherwise, the net that is inactive or heads a chain that dies out may exit from *s* to the left on the available track. The chain building and subsequent case determination is then performed with the other two nets.

Case 4.3 *s* is type 5

If *s* is of the form $\begin{array}{|c|c|c|}\hline a & & c \\ b' & & d \\ \hline\end{array}$ then *b*' must be externally inactive or head a chain of type 2 boards that dies out. If not a new *r* must be selected. Otherwise, *b*' and its chain are assigned to the available track to *s*'s left. Chain building and subsequent case determination is then performed with *c* and *d*.

If *s* is the form $\begin{array}{|c|c|c|}\hline a & & a \\ b' & & d \\ \hline\end{array}$ then at least one of *b* ' or *d* must be either externally inactive or head a chain of type 2 boards that dies out. If not a new *r* must be examined. Otherwise, the net that is externally inactive or heads a chain may exit from s to the left on the available track. Chain building and subsequent case determination is then performed with the other net.

Case 5. *a* is externally inactive and *b* is externally active and there is a single board *s* for the next board that requires net *b*

This case is analogous to case 4 and may be handled in a similar manner.

Case 6. *b* is externally inactive and *a* is externally active and there are several active boards that are part of net *a*

Suppose *q* is the previously assigned board.

To determine the next board *s* in the permutation *Bpd2* constructs the set *A* of active type 2 boards or type 3 boards that contain net *a* or active type 5 boards of the form $\begin{array}{|c|c|c|}\hline a & & a \\ b & & c \\ \hline\end{array}$.

The set *F* of active boards that are part of net *a* but are not in A is also constructed. Note, the set *F* may be empty.

The boards in *A* are the only boards in *Q'* that may be in the interior portion of the permutation spanning net *a*. The boards in *F* are those boards in *Q'* that must end net *a* if a density of 2 is to be achieved. As only one board can be last, if $|F| > 1$, a new *r* must be examined. Note that if *F* is empty then one of the boards in *A* ends net *a*.

For each active board s in *A* chain building is performed. If *s* has two chains then one exits left and the other exits right. If *s* has one chain then the net may exit either side. The result is a further combining of boards that can be readily transformed later to solve the original instance. Some of the boards in *A* may be forced to combine with each other due to the chain building. If this is so, it is easy to determine whether the resulting active board has density 2 while remaining in the interior. It is possible a board *u* in *A* has some externally active net *i* such that the connections from *i* do not form a chain, and hence more than one track is necessary. If so, *u* is placed in *F*.

In Figure 3 a partial BP instance is given. In the figure A equals $\{b_1, b_4, b_6\}$. From b_1 the chain (b_1, b_5, b_7, b_4) forces a combining of b_1 and b_4 in *A*. The result is that this portion of the permutation has density 2. From b_6 there are two chains (b_6, b_3) and (b_6, b_2). As these two chains do not combine with any other board in *A*, *A* is left at the end of this combining as $\{b_u, b_6\}$, where b_u is a new symbol representing the boards that were combined to form the subpermutation (b_1, b_5, b_7, b_4).

After the chain forcing and combining has been completed, the boards in *A* must have *a* as their only active net except for possibly one board. Otherwise, a permutation with density 2 is not possible and a new *r* must be examined by *Bpd2*. One board may have one other active net but it must be active in the board in *F* also if the permutation is to have density 2 and to avoid examining a new *r*. The boards in *A* can thus be assigned in arbitrary order immediately after q except for possibly one board that shares a second net with the board in F. This board must be the last board assigned from A. In Figure 3 the result is that b_u and b_6 only share net a so they can be laid out with either b_u or b_6 first. In the figure b_u occurs first.

If F is empty then all the boards must have been assigned and a permutation with density 2 has been found. This is also demonstrated in Figure 3. The solution for the reduced instance can be readily transformed to produce a solution for the original instance.

$$\begin{array}{ll} a = \{b_1, b_4, b_6, q\} & L \supseteq \{a, c, d, e, f\} \\ c = \{b_1, b_5, b_7\} & A = \{b_1, b_4, b_6\} \\ d = \{b_4, b_7\} & F = \phi \\ e = \{b_2, b_6\} & \\ f = \{b_3, b_6\} & \end{array}$$

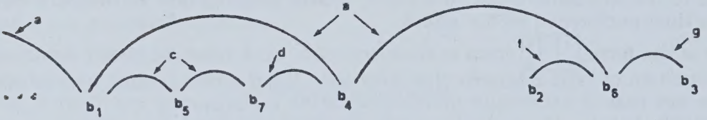


Figure 3 - The Combining Of Boards In A

If $|F|$ is greater than 1 then a new r must be selected.

If $|F|$ is 1 then let s be the element in F. There are two subcases that may allow the permutation to have density 2: the last board t in A assigned in the permutation has one connection with s, or t has two connections with s. Any more connections would imply that the density is greater than 2 and a new r must be examined.

Case 6.1 s only shares net a with t

If s only shares one connection to t then s may have at most three externally active nets. For the density to remain 2 with three active nets then one of these nets must form a chain that dies out, else a new r must be examined. The chain can exit left from as shown in Figure 4. If there are less than three active nets then the active nets may exit right and the inactive net may exit left if such a net exists. The chain building and subsequent case determination is continued with the nets that exit right.

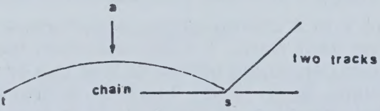


Figure 4 - t and s Combination

Case 6.2 s shares two nets a and b' with t

If s is type 2 or 3 then the nets a and b' enter s from the left, b' and the other net if it exists may also exit right. The net a at this point is inactive.

If s is type 4 then both a and b' enter left and must both become inactive and the remaining two nets must exit right. The chain building and subsequent case determination is performed on the nets that exit right. At this point a is guaranteed to be inactive but if b' is not inactive then a new r must be examined.

If s is type 5 then it must be of the form $\begin{array}{|c|c|c|}\hline a & & c \\ b' & & d \\ \hline\end{array}$ for the density to remain 2. Otherwise, a new r must be examined. The nets a and b' enter s from the left. The nets c and d exit from the right, the chain building and subsequent case determination is continued using these nets.

Case 7. a is externally inactive and b is externally active and there are several active boards that are part of net b

This case is analogous to case 6 and may be handled in a similar manner.

Thus a description of how to process a given r has been given. If the result is a permutation with density 2, all that remains is transforming the solution to one for the original instance.

3.4.2. Transforming The Permutation

A permutation for the reduced instance may be readily transformed to be a permutation for the original instance. The details appear in [COHO82a].

3.5. Run Time Analysis

A detailed complexity analysis of Bpd2 shows that it has a run time of $O(n^3)$.

4. Conclusions

The Board Permutation problem is an important problem in the design automation of digital systems. The Board Permutation problem has been previously shown to be NP-hard. Therefore, it is unlikely that there are efficient algorithms to the solve this problem. We have shown however, that there are efficient algorithms to determine whether the optimal density of an instance is 1 or 2. If the optimal density is 1 or 2 then a permutation with optimal density is also returned. It remains unknown whether there is an algorithm that determines in $O(n^{ck})$ time if a Board Permutation instance has density k.

5. Bibliography

[ADOL72] Adolphson, D. and T.C. Hu, "Optimal Linear Ordering", *SIAM J. Appl. Math.*, Nov. 1972, pp. 403-423.

[AHO74] Aho, A.V., J.E. Hopcroft, and J.D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA, 1974.

[CEDE74] Cederbaum, I., "Optimal Backboard Ordering through the Shortest Path Algorithm", *IEEE Trans. Circuits and Systems*, Sept. 1974, pp. 626-632.

[COHO82a] Cohoon, J., and S. Sahni, "Exact algorithms for the board permutation problem", Dept. of Computer Science, Univ. of Minnesota, Technical Report 82-15.

[COHO82b] Cohoon, J., and S. Sahni, "Heuristics for the Board Permutation Problem", Dept. of Computer Science, Univ. of Minnesota, Technical Report 82-16.

[GARE79] Garey, M.R., and D.S. Johnson, *Computers and Intractability - A Guide to the Theory of NP-completeness*, W.H. Freeman and Co., San Francisco, CA, 1979.

[GOTO77] Goto S., I. Cederbaum, and B.S. Ting, "Suboptimal Solution of the Backboard Ordering with Channel Capacity Constraint", *IEEE Trans. Circuits and Systems*, Nov. 1977, pp. 645-652.

[GURA81] Gurari, E.M., and I.H. Sudborough, "Cutwidth Problems in Graphs", 19th Annual Allerton Conference on Communication, Control, and Computing, 1981, pp. 752-761.

[HORO78] Horowitz, E. and S. Sahni, *Fundamentals of Computer Algorithms*, Computer Science Press, Potomac, MD, 1978.

[RUTM64] Rutman, R.A., "An Algorithm for Placement of Interconnected Elements Based on Minimum Wire Length", *Proc. SJCC*, 1964, pp. 477-491.

[SANG75] Sangiovanni-Vincentelli, A. and M. Santomauro, "A Heuristic Guided Algorithm for Optimal Backboard Ordering", presented at *13th Annual Allerton Conference on Circuit and System Theory*, 1975.

[STEI61] Steinberg, L., "The Back Board Wiring Problem: a Placement Algorithm", *SIAM Rev.*, Jan, 1961, pp. 37-50.

[TODD81] Todd, L.F., D.J. Gilbert, J.M. Hansen, R.J. Anderson, S.V. Pantula, A.K. Biyani, and J.L. Barron, "CGAL - A Multi Technology Gate Array Layout System", *ACM - IEEE 19th Annual Design Automation Conference*, 1982.

USING OVERLAP AND MINIMIZING CONTACT POINTS IN CHANNEL ROUTING

SUSANNE E. HAMBRUSCH
Department of Computer Science
Purdue University
West Lafayette, IN 47907.

Abstract. We study the effect of overlap on the channel width and the number of contact points needed for channel routing problems. We present a lower bound on the channel width for the k-layer model with k-fold overlap, $k \geq 2$.

Summary. The *Channel Routing Problem* (CRP) is a wiring problem in integrated circuit design that has received much attention recently ([BB], [D], [DKSSU], [PL], [RBM]). An important and extensively studied measure in channel routing is the channel width, but, besides minimizing the channel width, an efficient algorithm should also minimize the number of contact points. Since the technological obstacles to using more than two layers and/or a small amount of overlap have decreased, it is furthermore important to understand the relationship between the number of layers and the channel width, and how overlap affects channel width and the number of contact points.

In the 2-terminal CRP we are given n nets (p_i, q_i), where p_i and q_i are on opposite sides of the channel. A solution to the CRP consists of the channel width t and the wires connecting the two terminals of each net. In our models each layer can be used for horizontal and vertical wires. Wires on different layers can cross or share a corner (i.e., form a knock-knee), but are only allowed to overlap in the overlap model. (For a detailed description of the model see [H1].)

Solutions presented for the 2-layer model without overlap in [BB] and [RBM] achieve a channel width of $2d-1$ and use $O(dn)$ contact points when solving CRP of density d. (The density d is the maximum over all x of the number of nets (p, q) for which $p < x < q$ or $q < x < p$.) Leighton has shown that there exist CRPs of density d that require $2d-1$ tracks [L]. We first describe an algorithm for one-sided CRPs in the 2-layer model without overlap that uses $2d-1$ tracks and at most $4n$ contact points. A one-sided CRP consists either of right and trivial nets (right CRP), or of left and trivial nets (left CRP). A net (p_i, q_i) is *right* if $p_i < q_i$, *left* if $p_i > q_i$, and *trivial* if $p_i = q_i$. As in [PL], we distinguish between the *wire layout* and the *wiring* of a CRP. Informally, the wire layout describes the path of the wires without considering their assignment to the layers, while the wiring gives the assignment of wires to the layers. In the 2-layer model we call the wires running on the top (bottom) layer T- (B-)wires.

The obvious wire layout for right CRP is to assign each run (i.e., the maximal sequence of continuous right nets) to one of d tracks and to produce a knock-knee when a net ends and a new one starts up. If the wiring is then obtained by using $d-1$ interleaved tracks to accommodate necessary contact points, up to dn contact points are needed ([BB], [RMB]). It can be shown that in order to minimize the contact points the shape of the wired nets has to be *different* from the 'obvious' shape. Our algorithm uses d tracks for the horizontal wires and $d-1$ interleaved tracks for contact points. The wiring is determined column by column, from left to right. We try to run all horizontal wires as T-wires and all vertical wires as B-wires. This is not always possible: when a net reaches its final column as a T-wire, the wire immediately continuing on this track has to start off as a B-wire. The algorithm removes such a B-wire from the track within the next two columns by continuing it on another track as either a T-wire, or as a B-wire (if it again participates in a knock-knee). There are at most 3 horizontal B-wires between any two columns. Furthermore, a horizontal T-wire changes into a B-wire only when its corresponding net reaches the final column (in some cases it will be the column before the final one). Since each wired net contains at most 4 contact points, and the total number of contact points is at most $4n$.

Lower Bounds on the channel width for this and for another, more restrictive, model have been obtained by considering only right (or left) CRPs ([BR], [L]). It is not known whether $2d-1$ tracks and $O(n)$ contact points can be achieved simultaneously, and right (or left) CRPs alone are not powerful enough to prove it cannot be done. Our algorithm can be modified to solve *weak* CRPs using $2d-1$ tracks and $O(n)$ contact points. A weak CRP is a CRP that contains right and left nets with the following restriction: in all columns where one terminal belongs to a right and one to a left net, we either have always a right net end and a left net begin, or a right net begin and a left net end. If we remove this restriction the algorithm fails since the ordering of the wires in the channel is now crucial, and a predetermined ordering can be destroyed when changing B-wires into T-wires.

When a vertical overlap of length 1 is allowed between two nets we can route in a channel of $2d-1$ tracks using at most $3n$ contact points. We thus trade knock-knees and $O(dn)$ contact points for slightly more overlap and $O(n)$ contact points. An algorithm given in [RBM] uses vertical overlap of length $O(d)$ to achieve $O(n)$ contact points. Our vertical unit overlap algorithm consists of a wire layout and a wiring step. The wire layout on d tracks is found by using the algorithm given in [PL]. The general idea of the wiring is to replace each knock-knee by a slip and a vertical unit overlap in the next column. Since this overlap can cause problems when processing the next column, additional wiring rules are needed.

For the k-layer model with k-fold overlap $\lceil d/k \rceil$ is the trivial lower bound on the channel width. Using a counting argument we can show that for a large class of CRPs at least $\lceil d/(k-1) \rceil$ tracks are needed even when horizontal and vertical overlap of arbitrary length are allowed. Thus, at least d tracks are needed for the 2-layer model with overlap. The best upper bound known achieves $7/4 . d + c$ tracks, for some constant c [B]. For the details of the algorithms and the lower bound see [H1]. Channel routing algorithms for overlap models with more than 2 layers are given in [H2].

References

[B] D.J. Brown, private communication, 1983.

[BB] T. Bolognesi, D.J. Brown, 'A Channel Routing Algorithm with Bounded Wire Length', unpublished manuscript, 1982.

[BR] D.J. Brown, R.L. Rivest, 'New Lower Bounds for Channel Width', *Proc. of the CMU Conf. on VLSI Syst. and Comp.*, pp 153-159, 1981.

[D] D.N. Deutsch, 'A Dogleg Channel Router', *Proceedings of the 13th IEEE Design Automation Conf.*, pp 425-433, 1976.

[DKSSU] D. Dolev, K. Karplus, A. Siegel, A. Strong, J.D. Ullman, 'Optimal Wiring between Rectangles', *13th ACM SIGACT Conf.*, 1981.

[H1] S.E. Hambrusch, 'Minimizing Contact Points and Using Overlap on Two Layers', Techn. Report, CSD-TR-455, Purdue University, 1983.

[H2] S.E. Hambrusch, 'Channel Routing Algorithms for Overlap Models', in preparation.

[L] F.T. Leighton, 'New Lower Bounds for Channel Routing', unpublished manuscript, 1981.

[PL] F.P. Preparata, W. Lipski, 'Three Layers are enough', *Proceedings of the 23rd Annual IEEE FOCS Conf.*, pp 350-357, 1982.

[RBM] R.L. Rivest, A.E. Baratz, G. Miller, 'Provably Good Channel Routing Algorithms', *CMU Conf. on VLSI Syst. and Comp.*, 1981.

On Simulation and Interrelations between Boolean and VLSI circuits
(Extended Abstract)

Alok Aggarwal

Electrical Engineering and Computer Science
The Johns Hopkins University
Baltimore,MD 21218.

Introduction

To understand computational complexity, some effort has been spent in identifying relationships among various models of computation. Strong relationships between Turing machine resource-based complexity and circuit complexity have been established. In particular, it has been shown that Turing machine time and space correspond to circuit size and depth respectively, and these relationships even hold simultaneously [Bo77, Pi79, Ru80]. We seek to derive similar relationships among the resources of VLSI and Boolean circuits. Some results have been obtained in this direction by Savage [Sa81, Sa82] for semelective and multilective planar circuits. He proves that the planar circuit complexity, C_p, bounds the *area**(*time*)^2 product from below, for any function f, and he also determines the planar circuit complexity of some well known functions. Our results are summarized below :

[1] The notion of generalized planar complexity, C_p^*, is introduced and relations between area and period in the VLSI model and C_p^* and C_p in the Boolean circuit model [Sa81, LT79] are established. Many Boolean functions for which C_p and C_p^* are $\Theta(2^n)$ are shown to exist, thereby improving a known bound. It is shown that some results which have been stated for planar circuit complexity in [Sa81] seem to work for the generalized planar complexity only.

[2] If a chip H of area A and period P computes a function f then so do chips H^* and H^{**} of areas $O(A(P/P_1)^2)$ and $O(A^{3/2}(P/P_1))$ and period P_1 for $(P_1 \leqslant P)$. This result is optimal in that there exist functions (such as transitive functions [Vu80]) which can only be computed in a period P_1 by a chip that has either $\Omega(A(P/P_1)^2)$ area or has $\Omega(A^{3/2}(P/P_1))$ area.

This extended abstract is divided into four sections. Section I briefly describes various models, introduces the notion of generalized planar complexity, and reviews some basic definitions. Section II relates various measures for Boolean circuits. Section III discusses the simulation of one VLSI circuit by another and derives relationships between area and period in VLSI circuits and different measures in Boolean circuits. Section IV presents a discussion and some associated unresolved problems. Detailed proofs and related results will appear in the final draft of the paper.

I. The Models

[a] VLSI Model

The underlying model used is a generalization of Thompson's model [Th79], and this is described below briefly:

A VLSI graph, H, consists of nodes and wires laid in a rectangular grid such that the nodes of H correspond to the nodes of the grid, two wires overlap only at a grid node, and at most four wires are incident on a node of H. The nodes process the information and communicate with each other through the interconnecting wires. Some nodes, called I/O ports, are used to communicate with the external world. Each input is read once. The inputs and outputs must be read and delivered according to a predetermined sequence of time instants and on prespecified locations which do not depend on any data instance. Also, the computation performed by H is independent of the sequence of data instances. In other words, I/O schedule is semelocal and where and when oblivious [Sa81,LS81].

The unit of information is either a word (as in sorting or merging) or a bit (as in cyclic shift). If the unit is a word then a processor consists of a constant number of registers and a processing unit which performs simple arithmetic and Boolean operations in unit time. If the unit is a bit then each node consists of a constant number of flip flops and logic gates. If the unit is a bit (a word) then the model is called the bit model (the word model, respectively). Unless stated otherwise, we will be dealing with the bit model only. The delay along every wire is taken to be unity. The area, A, of the chip is the minimum area of its bounding rectangle; period, P, is the minimum time interval elapsed between the feeding of the first input bits of two consecutive data instances; and the worst case time, T, is the time elapsed between the input of the first input bit and the emergence of the last output bit for the most time consuming instance.

[b] The Boolean Circuit Model

A Boolean function f : $\{0,1\}^n \longrightarrow \{0,1\}^m$ has combinational complexity $C(f)$ if the minimal number of logic elements in any circuit which computes f is $C(f)$. This function has a planar combinational complexity $C_p(f)$ if the minimum number of logic elements in a planar logic circuit for computing f is $C_p(f)$. Also f has a delay $D(f)$ if the minimum delay in any circuit which computes f is $D(f)$. The generalized planar complexity $C_p^*(f)$ is the minimal number of logic elements and crossings in any embedding of any logic circuit that computes f.

[c] The Synchronous, two-bounded Boolean circuit model

Since the predecessors of each vertex are ordered and the Boolean circuit is acyclic, we can rank all the logic elements such that the source nodes are assigned a rank zero and if u is predecessor of w then the rank of w equals (1 + rank of u). A synchronous, combinational circuit is a Boolean circuit in which the predecessors of every logic element have the same rank. Furthermore, a synchronous Boolean circuit is two-bounded if the out-degree of every node (including the source nodes) is bounded by two. Then, analogous to the combinational complexity, planar circuit complexity and the generalized planar complexity, we can also define the synchronous, combinational complexity $C_s(f)$; the synchronous, planar circuit complexity $C_{p,s}(f)$; and the synchronous, generalized planar complexity $C_{p,s}^*(f)$. An excellent introduction to the Boolean circuit model can be found in [Sa75]. In this paper, by synchronous Boolean circuits we mean synchronous, two-bounded Boolean circuits. $C_p(f)$ was introduced by Savage [Sa81] and by Lipton and Tarjan [LT80] and they established that $C_p(f)$ for any superconcentrator is $\Omega(n^2)$.

Notation and Definitions

Following Leiserson [Le80], we introduce the notion of separator for general graphs as follows.

[Le80] Let S be a class of graphs closed under the subgraph relation i.e. if G belongs to S and if G' is a subgraph of G then G' also belongs to S. A $\lambda(n)$-separator theorem is a theorem of the following form :

There exist constants c_s and α_s where $0<\alpha_s<1/2$ and $c_s>0$ s.t. if G is an n vertex graph in S then by removing at most $c_s\lambda(n)$ edges of G, G can be partitioned in two disjoint subgraphs G_1 and G_2 having αn and $(1-\alpha)n$ vertices where $\alpha_s \leqslant \alpha \leqslant 0.5$.

This notion can be easily extended to Boolean circuits which belong the class of directed, acyclic graphs.

II. Relations among various resources

Observation 2.1: We note that the inequality $C_p^*(f) \leqslant C_p(f)$ holds because the number of crossings in a planar circuit is zero. However, it is not clear whether $C_p^*(f) = \Theta(C_p(f))$. In fact, the following statement due to Lipton and Tarjan [LT80], a variation of which is also used by Savage [Sa81], is shown to be false.

[LT80] " Any circuit can be converted into a planar circuit by the following steps. First, embed the circuit in the plane allowing the edges to cross, if necessary. Next replace each pair of crossing edges by the crossover circuit illustrated in figure 1."

We demonstrate the invalidity of the above statement. Figure 2 illustrates an embedding of a logical circuit that computes a function of four variables and figure 3 is obtained by replacing each pair of crossing edges of figure 2 by a crossover circuit. Now, it is easy to observe that the embedding of figure 3 is planar but the graph is no longer acyclic.

Remarks:

[1] Though the statement in [LT80] need not be true, we can employ the basic argument to obtain a planar, acyclic circuit if we embed the logic circuit (possibly, nonplanar) s.t. an edge between levels i and j $(j>i)$ crosses an edge between levels k and l $(l>k)$ only if either $i \leqslant k < j$ or $k \leqslant i < l$. An embedding of this kind for the circuit of figure 2 is shown in figure 4.

[2] The inequalities of [LT80] and [Sa81] derived for $C_p(f)$ can also be stated in terms of the generalized planar complexity and some of these inequalities are given below:
(i) Any m-input, m-output superconcentrator satisfies $C_p^*(f) = \Omega(m^2)$.
(ii) If $f:\{0,1\}^n \longrightarrow \{0,1\}^m$ is w-separated or has a w-flow then $C_p^*(f) = \Omega(w^2)$.

Similar bounds can also be identified for the generalized planar complexity of multilective planar circuits [Sa82].

[3] Observation 2.1 shows that there is a flaw in the proof of theorem 4 of [Sa81]. Though we could not settle the truth of the statement of this theorem, we were able to establish similar results for $C_p^*(f)$ and these are stated in theorem 3.4.

[4] In a similar manner $C_{p,s}(f) \geqslant C_{p,s}^*(f)$ can be shown. Whether or not $C_{p,s}(f) = \Theta(C_{p,s}^*(f))$ also remains open. Various relations among different measures of Boolean circuits are listed in table 4.1.

Theorem 2.2: For any function $f:\{0,1\}^n \longrightarrow \{0,1\}^m$
(a) $C(f) \leqslant C_p^*(f) \leqslant 2C^2(f)$
and (b) if a Boolean circuit with C(f) elements satisfies a $\lambda(n)$- separator theorem then
(i) $C_p^*(f) = O[\lambda^2(n)]$ if $\lambda(n) = \Omega(n^\epsilon)$ for some $\epsilon > 0.5$.
(ii) $C_p^*(f) = O[C(f)]$ if $\lambda(n) = O(n^\epsilon)$ for $\epsilon < 0.5$.
(iii) $C_p^*(f) = O[C(f) \log C(f)]$ if $\lambda(n) = O(\sqrt{n})$.

Theorem 2.3 : For any function $f:\{0,1\}^n \longrightarrow \{0,1\}^m$,
$C(f) \leqslant C_p(f) \leqslant 6[C^2(f)]$.

Proof : We embed the logic circuit with C(f) nodes in the plane s.t. all the nodes that belong to the i-th level of the logic circuit are placed along the i-th vertical line in the embedding and any edge incident on nodes at level i and j with $j > i$ is embedded so that it crosses the vertical lines $(i+1), (i+2), \ldots, (j-1)$ only once. Then an edge between level i and level j $(j>i)$ crosses an edge between level k and level l $(l>j)$ only if either $i \leqslant k < j$ or $k \leqslant i < l$ and the total number of crossings does not exceed $C(f)^*(2C(f)-1)$. Now using Observation 2.1, we replace each crossing by three exclusive OR's and note that the total number of logic elements in the obtained circuit is no more than $C(f)+3C(f)^*(2C(f)-1)$.

Theorem 2.4 : For any $0<\delta<1$, a fraction of at least $1-2^{\delta m 2^n}$ of the Boolean functions have :

$C_p^*(f) \geqslant m2^n(1-\delta)/30$ and for all such functions $C_p(f) \leqslant 7m2^n$.

The upper bound was established by Savage [Sa81]. The lower bound on $C_p^*(f)$ is obtained by counting the total number planar embeddings with at most $C_p^*(f)$ logic elements and crossings. A result due to Tutte [Tu63] is used and this described below briefly :

Let M be a planar embedding with outer face F. Select one of the edges R of the bounding path of F and call it the root. The choice of a root converts M into a *rooted embedding*. Then from Tutte [Tu63], the total number of rooted embeddings with r edges equals $\{2^*(2r)!3^r/(r+2)!r!\}$. It is easy to see that the total number of planar embeddings is, therefore no more than $\{2^*(2r)!3^r/(r+2)!r!\}$ i.e. no more than 12^r.

III. Relations among VLSI resources and Boolean circuit measures

Theorem 3.1 : If a synchronous, two-bounded Boolean circuit computes a function $f:\{0,1\}^n \longrightarrow \{0,1\}^m$ and has $C_s(f)$ logic elements then there exists a VLSI circuit which computes f in $O[C_s^2(f)]$ area and has a unit period.

Proof : The proof of this is obtained by embedding the given Boolean circuit with at most $2C_s(f)$ edges in a rectangular grid and then noting that the resulting circuit is a VLSI circuit with a unit period and at most $4C_s^2(f)$ area. Following theorem 2.2, we can also incorporate the notion of a separator to obtain better bounds on the area of the VLSI chip. Consequently, if a synchronous, two-bounded Boolean circuit computes f and has a $\lambda(n)$-separator theorem then there exists a VLSI chip of area A and unit period s.t.

(a) $A = O[\lambda^2(n)]$ if $\lambda(n) = \Omega(n^\epsilon)$ and $\epsilon > 0.5$.

(b) $A = O[C_s(f) + n]$ if $\lambda(n) = O(n^\epsilon)$ and $\epsilon < 0.5$.

(c) $A = O[[C_s(f) + n]\log^2 [C_s(f) + n] + n]$ if $\lambda(n) = O(n^{0.5})$.

The proof of this can be obtained using Leiserson's approach [Le80] for laying the given circuit in a rectangular grid.

Theorem 3.2: From Savage [Sa81], we recall that if $f:\{0,1\}^n \longrightarrow \{0,1\}^m$ is computed in T steps by a VLSI chip that has area A in the bit model, then the inequality $AT \leqslant C(f)$ must be satisfied.

Proof : The proof of this is obtained by first constructing T copies of the chip, then breaking the existing loops in the copies and feeding the signals to the corresponding nodes in the next copy. It is easy to see that $AT \geqslant C_s(f) - nT$ must also be satisfied.

From Thompson [Th80], we recall Lemma 3.3. A variant of this has also been used by Savage [Sa81].

Lemma 3.3 [Thompson] : If a VLSI chip H of area A requires k layers of interconnection then there exists a chip $\hat{H}$ which requires at most $4Ak^2$ area in the VLSI model of Section I.

Theorem 3.4 : Let $f:\{0,1\}^n \longrightarrow \{0,1\}^m$ be computed in T steps by a chip H of area A in the bit model. Then the following relationships always hold:

(i) $AT^2 = \Omega(C_p^*(f))$.

(ii) $A^{3/2}T = \Omega(C_p^*(f))$.

(iii) $A^2T = \Omega(C_p(f))$.

Proof : Let l and b be the dimensions of the chip H with $l \geqslant b$ and $b \leqslant \sqrt{A}$. Then as in Observation 3.2, we create a stack of T chips and follow steps (a) or (b) given below to obtain the stated inequalities:

(a) Break the loops in a copy and feed these signals suitably to the corresponding position in the copy above. The new connections are made to run along the rectangular grid so that a cuboid of size at most $2l^* 2b^* 4T$ is obtained. This stack is now transformed in one of the following ways:

(i) Use Theorem 3.3 with k=4T. The area of the obtained chip is no more than $16l^*b^* 16T^2$ i.e. $C_p^*(f) \leqslant 256AT^2$.

(ii) Use Theorem 3.3 with k=2b. The area of the obtained chip is no more than $32lb^{2*} 4T$ i.e. $C_p^*(f) \leqslant 128A^{3/2}T$.

(b) Lay T copies of the chip H side by side so that the first and the T-th copies are the leftmost and the rightmost copies respectively. Again, break the loops and connect the signals suitably to the corresponding position in the next copy and then embed the obtained graph so that the embedding contains crossings between the wires joining nodes in copies i and (i+1) only, for $1 \leqslant i \leqslant T-1$. Since the total number of wires between the copies i and (i+1) is no more than 2A, the total number of crossings between these copies does not exceed $A(2A-1)$. Now, using observation 2.1, we convert this into a planar circuit and note that $C_p(f) \leqslant 3A(2A-1)T + AT$.

Remarks :

[1] From observations 2.4 and 3.4 it follows that for any $0 < \delta < 1$, a fraction of at least $1-2^{\delta m 2^n}$ Boolean functions have AT^2 and $A^{3/2}T = \Omega(m 2^{(n(1-\delta)})$. However, it is not clear whether these bounds are optimal within a constant. (See also section IV).

[2] Following the arguments of Observation 3.2 and 3.4, the following inequalities can be also shown.
(a) AT^2, $A^{3/2}T = \Omega(C_{p,s}^*(f) - nT)$.
(b) $A^2T = \Omega(C_{p,s}(f) - nT)$.

Theorem 3.5 : In the word model, if a chip H computes a function f in area A, period P and time T, then there exists a chip H' which computes f in area $O(A(P/P_1)^2)$, period P_1 and time O(T) for $P_1 \leqslant P$.

Proof : Roughly speaking, the basic idea used here is similar to that in the previous theorems. However, we use only (P/P_1) copies and each copy carries out the execution for P_1 steps (rather than one step as in theorem 3.4).

Corollary 3.6 : In the word model, if a chip H computes a function f in area A, period P and time T, then there exists a chip H^{**} with area $O(A^{3/2}(P/P_1))$, which computes the function f in period P_1 and time O(T) for $P_1 \leqslant P$.

Remarks : It is easy to see that theorems 3.5 and 3.6 are both tight in that, for any arbitrary function f, there cannot exist a chip H^{***} which computes f in period P_1, time O(T) and is of area $o(A(P/P_1)^2)$ or area $o(A^{3/2}(P/P_1))$. The proof of this statement can be obtained easily by noting that a rectangular mesh of $O(n)$ area sorts n numbers with $O(\sqrt{n})$ period and if a chip H^{***} of $o(n^2/P_1^2)$ area exists and sorts n numbers in period P_1 then $area^*(period)^2$ for H^{***} is $o(n^2)$ and thus H^{***} does not obey $AP^2 = \Omega(n^2)$ [Vu80]. A similar argument can be advanced if H^{***} sorts n numbers in $o(n^2/P_1))$ area and with a period P_1.

IV. Conclusion

[a] Discussion : In proving the lower bounds on area and time in terms of Boolean circuit measures, the main argument used was the unravelling of the VLSI circuit so that the resulting circuit is acyclic i.e. each node in the resulting circuit is used at most once in the computation of any data instance. This unravelling was achieved by constructing T copies of the VLSI chip and then removing the loops in each copy. It is our conjecture that, in fact, P copies are sufficient to obtain an acyclic circuit. Roughly speaking, this should be so because if a chip H has a period P then any node of H is used at most P times in the computation of any data instance. Consequently, replacing each node of H by its P copies and then connecting these nodes suitably should result in a Boolean circuit. Thus, we have :

Conjecture : If a function $f:\{0,1\}^n \longrightarrow \{0,1\}^m$ is computed by a VLSI chip H in area A and period P then the inequality $AP \geqslant C(f)$ must be satisfied. In fact, it was this conjecture which led to the discovery of Theorem 2.4.

Tables 4.1 and 4.2 summarize the interrelationships among the different measures of Boolean circuits and among the Boolean and the VLSI circuits, respectively. The possible improvement due to the separator theorem is not incorporated in these tables except in table 4.2(b) [Cf. Remark [1] of theorem 3.1].

[b] Extensions and Open Problems : Though we were able interrelate various resources of the two models, the following problems remain open :

(i) The optimality of the existential lower bound and the constructive upper bound has been demonstrated, within a constant factor, for $C_{p(f)}$ and $C_p^*(f)$ for the most complex Boolean functions on n variables. Lupanov [Lu73] has developed similar bounds for the unbounded degree, synchronous, combinational complexity for functions on n variables. However, there exist large gaps in the two bounds for $C_{p,s(f)}$ $C_{p,s}(f)$ and $area-(time)^2$, $area-(period)^2$ in Boolean and VLSI circuits, respectively. Also, no good bounds are known for the minimum area required for VLSI chips for computing for the most complex Boolean functions on n variables. The only known technique to prove lower bounds on area is due to Yao [Ya81] and a variation of this has been used by Baudet [Ba81] and by Savage [Sa81]. However, this technique seems to yield bounds which are no better than $\Omega(n)$.

(ii) It would be interesting to prove a result similar to theorem 3.5 for the bit model. It is easy to see that any generalization of theorem 3.5 in the bit model is no longer optimal because performing comparison and counting in a VLSI circuit increases the area, period and time of the resulting circuit by at least $O(logT)$.

$$\begin{gathered}C_p(f) \geqslant C_p^*(f) \geqslant C(f). \\ C_{p,s}(f) \geqslant C_{p,s}^*(f) \geqslant C_s(f). \\ C_{p,s}(f) \geqslant C_p(f) : C_{p,s}^*(f) \geqslant C_p^*(f). \\ C_{p,s}(f)\,,\, C_{p,s}^*(f)\,,\, C_p^*(f)\,,\, C_p(f) \leqslant 6C^2(f).\end{gathered}$$

Table 4.1 : Relations amongst various Boolean Circuit Measures.

Lower Bounds	Upper Bounds
$AT = \Omega(C(f))$ $AT = \Omega(C_s(f) - nT)$ $AT^2, A^{3/2}T = \Omega(C_p^*(f))$ $A^2T = \Omega(C_p(f))$	$AP^\alpha = O(C^2(f))$ for *any* $\alpha > 0$.
$AT^2\,,\, A^{3/2}T = \Omega(C_{p,s}(f) - nT)$ $A^2T = \Omega(C_{p,s}(f) - nT)$	$AP^\alpha = O(C_{p,s}^*(f)\log^2(C_{p,s}(f)))$ for *any* $\alpha > 0$.

Table 4.2 : Bounds on Area, Period and Time in terms of Boolean Circuit Measures.

Acknowledgements : The author thanks Prof. S. Rao Kosaraju of the Johns Hopkins University for funding this research from his NSF grant No. MCS 820-5167 and for his invaluable suggestions. He also acknowledges clarifying discussions with Prof. John E. Savage of Brown University as well as the help he received from Prof. Joseph O'Rourke and Dr. Kathleen Callaway of the Johns Hopkins University in preparing this manuscript.

References :

[Ba81] G. M. Baudet, "On the area required by VLSI circuits," Proceedings of the CMU conference on VLSI systems and computations edited by H. T. Kung, B. Sproull and G. Steele, CS Press, Maryland, Oct. 1981 pp. 100-107.

[Bo77] A. Borodin, "On relating time and space to size and depth," SIAM Journal of Computing, Vol. 6, No. 4, 1977, pp. 733-744.

[HS79] L. H. Harper and J. E. Savage, "Lower bounds on Synchronous combinational complexity," SIAM Journal of Computing, Vol. 8, No. 2, 1979, pp. 115-119.

[Le80] C. E. Leiserson, "Area-efficient graph layouts (for VLSI)," Proceedings of the 21st Annual IEEE Symposium on Foundations of Computer Science, Oct. 1980, pp. 270-281.

[LT79] R. J. Lipton and R. E. Tarjan, "A separator theorem for planar graphs," SIAM Journal of App. Math., Vol. 36, 1979, pp. 177-189.

[LT80] R. J. Lipton and R. E. Tarjan, "Applications of a planar separator theorem," SIAM Journal of Computing, Vol. 9, No. 3, August 1980.

[LS81] R. J. Lipton and R. Sedgewick, "Lower bounds for VLSI," Proceedings 13th Annual Symposium on the Theory of Computing, May 1981, pp. 300-307.

[Lu73] O. B. Lupanov, "On networks of functional elements with delays," Systems Theory Research, Vol. 23, 1973, pp. 455-461.

[Pi79] N. Pippenger, "On simultaneous resource bounds," Proceedings of 20th Annual IEEE Symposium on the Foundations of Computer Science, 1979, pp. 312-318.

[Ru80] W. L. Ruzzo, "Tree size bounded alternation," Journal of Computer Science and Systems, Vol. 21, 1980, pp. 218-235.

[Sa75] J. E. Savage,"The complexity of Computing," Chapter II, Wiley-Interscience, New York, 1976.

[Sa81] J. E. Savage,"Planar Circuit complexity and the performance of VLSI circuits," Proceedings of the CMU Coference on VLSI systems and computations, edited by H. T. Kung, B. Sproull and G. Steele, CS Press, Maryland, Oct. 1981, pp. 81-88.

[Sa82] J. E. Savage, "Bounds on the performance of multilective VLSI algorithm," Technical report No. CS-82-10, Dept. of Computer Science, Brown University, March 1982, pp. 1-28.

[Th79] C. D. Thompson, "Area-time complexity for VLSI," Proceedings of 11th Annual Symposium on the Theory of Computing, 1979, pp. 81-88.

[Th80] C. D. Thompson, "A Complexity theory for VLSI," PhD dissertation, Department of Computer Science, Carnegie Mellon University, 1980.

[Tu63] W. T. Tutte, "A census of planar maps," Canadian Journal of Math. , Vol. 15, pp. 249-279.

[Vu80] J. E. Vuillimin, "A Combinatorial limit to the computing power of VLSI circuits," Foundations of Computer Science, Nov. 1980, pp. 294-300.

[Ya81] A. C. Yao, "The entropic limitations on VLSI computations," Proceedings of the Thirteenth Annual Symposium on the Theory of Computing, 1981, pp. 308-311.

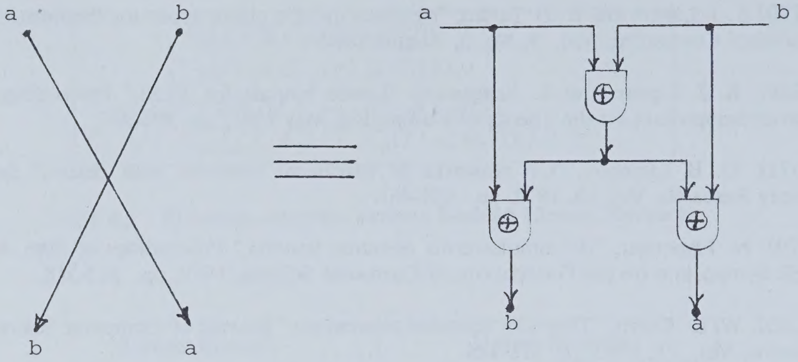
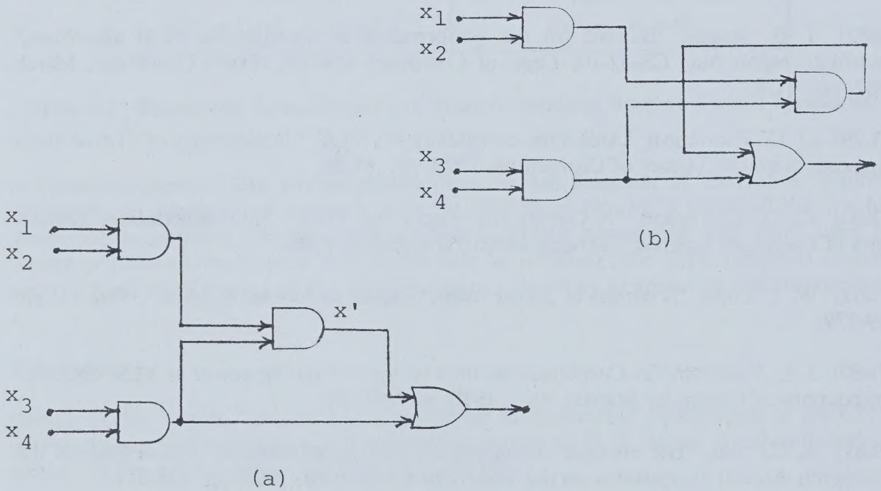


Fig. 1: Replacing Crossing Wires by a Planar Exclusive or Circuit.


x1
x2
x3
x4
x1
x2
x'
x3
x4

(b)

(a)

Fig. 2: (a) A Logic Circuit for a Function of Four Variables.
(b) An Embedding of the Logic Circuit of Fig. 2(a).

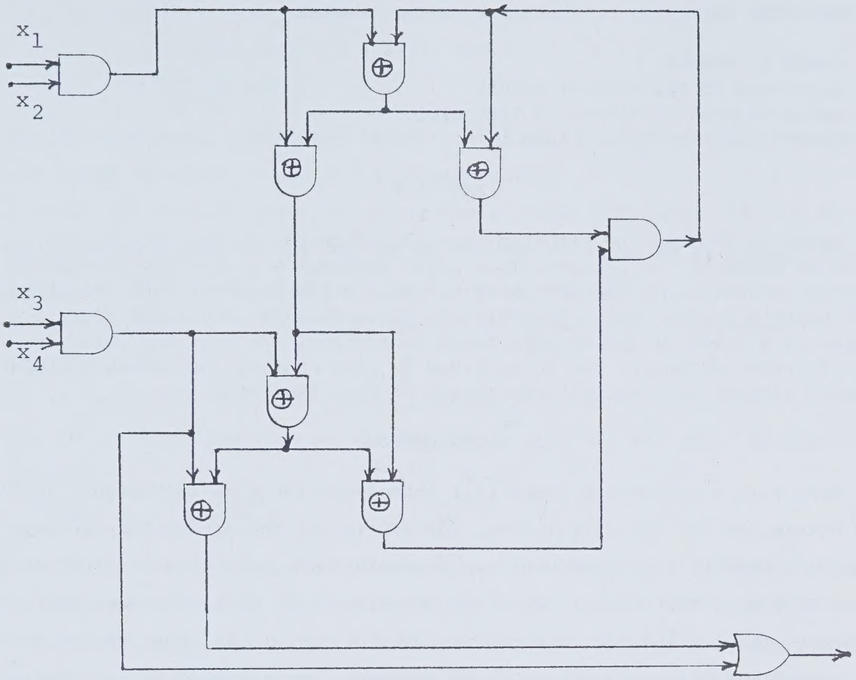
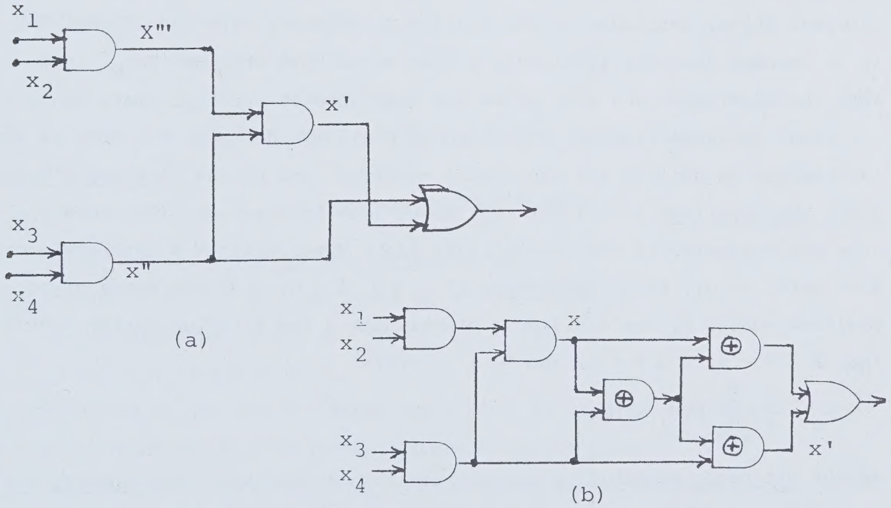


Fig. 3: Circuit Obtained Following the Steps of [LT80].


x1
x2
X'''
x'
x3
x4
x"
x1
x2
x
x3
x4
x'

(a)

(b)

Fig. 4: (a) An Embedding of the Logic Circuit of Fig. 2(a) Following Obs. 2.1.
(b) The Corresponding Planar, Acyclic Circuit.

A SIMPLIFIED TREATMENT OF WYNER'S WIRE-TAP CHANNEL

JAMES L. MASSEY
Institute of Telecommunications
Swiss Federal Institute of Technology
CH-8092 Zurich, Switzerland

ABSTRACT

Wyner introduced the wire-tap channel and found its capacity region, which he noted was not convex. This paper introduces a capacity pre-region that is convex, and whose convexity provides a simple proof that points outside Wyner's region cannot be achieved. It is further shown that Wyner's region is a union of simply structured subregions, and that any point in any of these subregions can be achieved by time-sharing the coding systems used to achieve the two critical points of that subregion.

I. INTRODUCTION

Wyner, in a pioneering paper [1], introduced the wire-tap channel (WTC) and determined its capacity region. The WTC is the one-sender two-receiver channel formed by the cascade of two discrete memoryless channels (DMC's) as shown in Fig.1. The channel structure coincides with that of a degraded broadcast channel [2], but the goal is very different. As shown in Fig.2, the sender seeks to transmit a binary symmetric source (BSS) to the legitimate receiver over the main channel (DMC_M) while denying this information as much as possible to the wiretapper, who obtains a noisy version of the received signal available to the legitimate receiver over the channel DMC_W. It is assumed that the legitimate sender knows both DMC_M and DMC_W, as does also the wiretapper who also knows the code used by the legitimate sender.

Wyner characterized the situation of Fig.2 by the rate, $R = K/N$, at which information is sent to the legitimate receiver, and by the wiretapper's per digit equivocation, $\Delta = H(U^K|Z^N)/K$, about this information. (Our notation here and hereafter is that of Gallager [3].) Wyner defined a rate-equivocation pair, (r,d), to be achievable if $r \geq 0$, $d \geq 0$, and, for every given positive number ε, one can find a blocklength N and a coding system such that $R \geq r - \varepsilon$, $\Delta \geq d - \varepsilon$, and $P_b \leq \varepsilon$, where

$$P_b = \frac{1}{K} \sum_{i=1}^{K} P(\hat{U}_i \neq U_i) \qquad (1)$$

is the bit error probability for the legitimate receiver. The capacity region, $\mathcal{C}$, is the set of all achievable rate-equivocation pairs. Wyner proved that

$$\begin{aligned} \mathcal{C} = \{ (r,d) : r \geqslant 0,\ 0 \leqslant d \leqslant 1, \text{ and there exists} \\ p_o(x) \text{ such that } r \leqslant I_o(X;Y) \text{ and} \\ rd \leqslant I_o(X;Y|Z) \}, \end{aligned} \tag{2}$$

and he noted that, unlike the capacity regions of all previously considered multi-user channels, $\mathcal{C}$ was not a convex region.

Our goal in this paper is to provide a simplified proof that (2) gives the capacity region of the WTC. We do this by introducing a *capacity pre-region*, $\mathcal{W}$, that uniquely determines $\mathcal{C}$ and vice-versa, but that is convex. The convexity of $\mathcal{W}$ is then exploited in the proof that $\mathcal{C}$ is indeed the capacity region.

We shall have need for *Fano's inequality* [3, p.79] which asserts that when U^K is estimated from an observation W^N then the bit error probability satisfies

$$h(P_b) \geqslant \frac{1}{K} H(U^K | W^N), \tag{3}$$

where $h(p) = -p \log p - (1 - p) \log (1 - p)$ is the binary entropy function. We shall also make use of the result, which was not available to Wyner, that for the WTC, *both* $I(X;Y)$ *and* $I(X;Y|Z)$ *are concave functions of the input probability distribution* P_X, as shown by Leung-Yan-Cheong [4] with an elegantly simple argument. Finally, we shall need the fact that, for the WTC,

$$\begin{aligned} I(X;Y|Z) &= H(X|Z) - H(X|YZ) \\ &= H(X|Z) - H(X|Y) \\ &= I(X;Y) - I(X;Z), \end{aligned} \tag{4}$$

as follows from the fact that X,Y,Z is a Markov Chain.

II. THE CAPACITY PRE-REGION

We define the *capacity pre-region*, $\mathcal{W}$, as

$$\begin{aligned} \mathcal{W} = \{ (r,\gamma) : r \geqslant 0,\ \gamma \geqslant 0 \text{ and there exists} \\ p_o(x) \text{ such that } 0 \leqslant r \leqslant I_o(X;Y) \\ \text{and } 0 \leqslant \gamma \leqslant I_c(X;Y|Z) \}. \end{aligned} \tag{5}$$

A comparison of (2) and (5) shows that (r,γ) is in $\mathcal{W}$ if and only if $(r,r\gamma)$ is in $\mathcal{C}$. Moreover, $\mathcal{W}$ is a convex region as we now prove.

Suppose that (r_1,γ_1) and (r_2,γ_2) are pairs in $\mathcal{W}$ corresponding to $p_1(x)$ and $p_2(x)$, respectively. We must show that for every θ, $0 \leqslant \theta \leqslant 1$, the pair $(r,\gamma) = (\theta r_1 + (1-\theta) r_2,\ \theta\gamma_1 + (1-\theta)\gamma_2)$ is also in $\mathcal{W}$. Clearly, $r \geqslant 0$

and $\gamma \geq 0$. We now choose

$$p_o(X) = \theta p_1(x) + (1-\theta)p_2(x).$$

By the concavity of $I(X;Y)$ and $I(X;Y|Z)$, it follows that

$$\begin{aligned} I_o(X;Y) &\geq \theta I_1(X;Y) + (1-\theta)I_2(X;Y) \\ &\geq \theta r_1 + (1-\theta)r_2 = r \end{aligned}$$

and

$$\begin{aligned} I_o(X;Y|Z) &\geq \theta I_1(X;Y|Z) + (1-\theta)I_2(X;Y|Z) \\ &\geq \theta\gamma_1 + (1-\theta)\gamma_2 = \gamma, \end{aligned}$$

which proves that (r,γ) is indeed in $\mathcal{W}$.

From (5), we see that every input probability distribution, $p_o(x)$, determines the rectangular subregion of $\mathcal{W}$ shown in Fig. 3(a). Moreover, this rectangular region determines, and is determined by, the subregion of $\mathcal{C}$, corresponding to $p_o(x)$, shown in Fig.3(b). $\mathcal{W}$ is the union of all subregions as in Fig.3(a), just as $\mathcal{C}$ is the union of those in Fig.3(b). The interesting segment of the boundary of the capacity subregion of Fig.3(b) occurs over the rate interval

$$I_o(X;Y) - I_o(X;Z) \leq r \leq I_o(X;Y) \tag{6a}$$

for which the equivocation is given by the hyperbola

$$rd = I_o(X;Y) - I_o(X;Z) \tag{6b}$$

as follows from (2) and (4).

III. THE NECESSITY ARGUMENT

We now wish to show that, for (R,Δ) to be achievable, a necessary condition is that (R,Δ) be in $\mathcal{C}$ or, equivalently, that $(R,R\Delta)$ be in $\mathcal{W}$. The following result is the key to this demonstration.

Lemma 1: For the coding scheme of Fig.2, there exists $p_o(x)$ such that

$$I(X^N;Y^N) \leq NI_o(X;Y) \tag{7a}$$

and

$$I(X^N;Y^N|Z^N) \leq NI_o(X;Y|Z). \tag{7b}$$

Remark: Because of the non-negativity of mutual information, Lemma 1 is equivalent to the statement that (r,γ) is in $\mathcal{W}$, where $r = I(X^N;Y^N)/N$ and $\gamma = I(X^N;Y^N|Z^N)/N$.

Proof: By the well-known inequality [3, p.75] for a DMC,

$$\frac{1}{N}I(X^N;Y^N) \leq \sum_{i=1}^{N} \frac{1}{N} I(X_i;Y_i). \tag{8}$$

Thus, defining

$$p_o(x) = \sum_{i=1}^{N} \frac{1}{N} p_i(x) \tag{9}$$

where $p_i(x)$ is the probability distribution for X_i, we have by the concavity of $I(X;Y)$ that

$$I_o(X;Y) \geqslant \sum_{i=1}^{N} \frac{1}{N} I(X_i;Y_i). \tag{10}$$

Combining (8) and (10) now gives (7a).

By the concavity also of $I(X;Y|Z)$ for the WTC, we see that an entirely similar argument will give (7b), provided we can establish the inequality corresponding to (8), namely

$$I(X^N;Y^N|Z^N) \leqslant \sum_{i=1}^{N} \frac{1}{N} I(X_i;Y_i). \tag{11}$$

Inequality (11) was proved by Wyner [1] as follows:

$$\begin{aligned} I(X^N;Y^N|Z^N) &= H(Y^N|Z^N) - H(Y^N|X^N Z^N) \\ &= \sum_{i=1}^{N} \left[H(Y_i|Y^{i-1}Z^N) - H(Y_i|Y^{i-1}X^N Z^N)\right] \\ &\leqslant \sum_{i=1}^{N} \left[H(Y_i|Z_i) - H(Y_i|Y^{i-1}X^N Z^N)\right] = \\ &= \sum_{i=1}^{N} \left[H(Y_i|Z_i) - H(Y_i|X_i Z_i)\right] \\ &= \sum_{i=1}^{N} I(X_i;Y_i|Z_i), \end{aligned}$$

where the fact that $H(Y_i|Y^{i-1}X^N Z^N) = H(Y_i|X_i Z_i)$ follows from the fact that, given X_i and Z_i, Y_i has no further dependence on any of the digits X, Y, Z as can be seen from Fig.2.

We now prove the following result, of some independent interest, which immediately implies that if (R,Δ) is achievable, then $(R,R\Delta)$ is in $\mathcal{W}$.

Theorem 1: For the coding scheme of Fig.(2),

$$h(P_b) \geqslant 1 - I_o(X;Y)/R \tag{12a}$$

and

$$R\Delta \leqslant I_o(X;Y|Z) + Rh(P_b), \tag{12b}$$

where $p_o(x)$ is defined by (9).

Proof: We begin by noting that

$$\begin{aligned} H(U^K|Y^N) &= H(U^K) - I(X^N;Y^N) \\ &= K - I(X^N;Y^N) \\ &\geqslant K - NI_o(X;Y) \end{aligned}$$

where we have used the BSS entropy property and inequality (7a). Dividing by K and using (3) now gives (12a).

Next, we note that U^K, Y^N, Z^N form a Markov chain so that

$$\begin{aligned} I(U^K;Y^N|Z^N) &= H(U^K|Z^N) - H(U^K|Y^N Z^N) \\ &= H(U^K|Z^N) - H(U^K|Y^N). \end{aligned}$$

Again invoking Fano's inequality (3), we have thus

$$\begin{aligned} h(P_b) &\geqslant \frac{1}{K} H(U^K|Y^N) = \\ &= \frac{1}{K}\left[H(U^K|Z^N) - I(U^K;Y^N|Z^N)\right] \\ &= \frac{1}{K}\left[K\Delta - I(U^K;Y^N|Z^N)\right] \\ &\geqslant \Delta - \frac{1}{K} I(X^N,Y^N|Z^N) \\ &\geqslant \Delta - \frac{N}{K} I_o(X;Y|Z) \end{aligned}$$

where the last inequality follows from (7b), and where the preceding inequality follows from the fact that U^K, X^N, Y^N, Z^N form a Markov chain. Multiplying by $R = K/N$ and rearranging now gives (12b).

IV. THE SUFFICIENCY ARGUMENT

It remains to show that if (R,d) is in $\mathcal{C}$, then (R,d) can be achieved. But our discussion in Section II showed that if (R,d) is in $\mathcal{C}$, then (R,d) is also in the capacity subregion of Fig.3(b) for some $p_o(x)$. Hence, it suffices to show that if (R,d) is a point in such a subregion, then (R,d) can be achieved. It is obvious that if we can achieve any point on the outer boundary (that part of the boundary not coinciding with the coordinate axes), then we can achieve any point in the subregion. The two critical points on this outer boundary are the endpoints of the hyperbolic segment. It is obvious that if we can achieve these two points, we can also achieve the points on the linear portions of the outer boundary. The following result, however, is more surprising.

Theorem 2: Any point on the hyperbolic segment of the boundary of the capacity subregion in Fig.3(b) can be achieved by time-sharing of the coding schemes that achieve the two endpoints of this segment, (r_1,d_1) and (r_2,d_2).

Proof: Let (r,d) be the pair to be achieved; then $r = (1-\lambda)r_1 + \lambda r_2$ where $0 < \lambda < 1$. Without loss of essential generality, we may suppose that λ is a rational number, say $\lambda = m/n$.

Let $\varepsilon > 0$ be specified. Because (r_1,d_1) and (r_2,d_2) are achievable, we can find coding schemes with the same blocklength N_{12} that achieve these points in the manner

$$P_{b1} \leqslant \varepsilon \text{ and } P_{b2} \leqslant \varepsilon \tag{13a}$$

$$r_1 \geqslant R_1 \geqslant r_1 - \delta \text{ and } r_2 \geqslant R_2 \geqslant r_2 - \delta \tag{13b}$$

$$\Delta_1 \geqslant d_1 - \delta \text{ and } \Delta_2 \geqslant d_2 - \delta \tag{13c}$$

where δ satisfies $0 < \delta \leqslant \varepsilon$ and will be specified later. Now consider the "time-shared" coding system of blocklength $N = n\,N_{12}$ such that the first and second coding schemes are used $n - m$ and m times, respectively, in one block. Note that fractions $1 - \lambda$ and λ of the N uses of the WTC are made with the first and second coding system, respectively.

The bit error probability of the time-shared system certainly satisfies

$$P_b \leqslant \max\,(P_{b1},P_{b2}) \leqslant \varepsilon \tag{14a}$$

as required. The rate of the time-shared system is

$$R = (1-\lambda)R_1 + \lambda R_2$$

so that from (13b) we have

$$r \geqslant R \geqslant r - \delta \geqslant r - \varepsilon \tag{14b}$$

as desired. Finally, we note that the fraction of information bits transmitted via the first and second coding system is $(1-\lambda)R_1/R$ and $\lambda R_2/R$, respectively. Thus, the wiretapper's equivocation for the time-shared system is

$$\begin{aligned}\Delta &= (1-\lambda)R_1\Delta_1/R + \lambda R_2\Delta_2/R \\ &\geqslant (1-\lambda)(r_1-\delta)(d_1-\delta)/r + \lambda(r_2-\delta)(d_2-\delta)/r \\ &\geqslant (1-\lambda)(r_1 d_1 - r_1\delta - d_1\delta)/r + \lambda(r_2 d_2 - r_2\delta - d_2\delta)/r.\end{aligned} \tag{15}$$

But now we note that (r,d), (r_1,d_1) and (r_2,d_2) are all on the hyperbolic segment of the capacity subregion boundary so that (6b) implies

$$rd = r_1 d_1 = r_2 d_2. \tag{16}$$

Using (16) in (15) and noting that $d_1 \leqslant 1$ and $d_2 \leqslant 1$ now gives

$$\Delta \geqslant d - (r+2)\delta/r \geqslant d - \varepsilon \tag{14c}$$

as desired, provided we ensure that the free parameter δ also satisfies $\delta \leqslant r\varepsilon/(r+2)$. Equations (14a) - (14c) show that the time-shared system achieves (r,d).

To complete the proof that C is indeed the capacity region of the WTC, it remains only to show that the rate-equivocation pairs of Theorem 2, namely

$$(r_1,d_1) = (I_o(X;Y),\ 1 - I_o(X;Z)/I_o(X;Y)) \tag{17a}$$

and

$$(r_2,d_2) = (I_o(X;Y) - I_o(X;Z),\ 1) \tag{17b}$$

can be achieved.

The pair (r_1,d_1) is easy to obtain. A standard random coding scheme (cf. [3,pp.131-141]) in which each digit of each codeword is chosen independently according to $p_o(x)$ will, for codes of rate $R = r_1 - \delta = I_o(X;Y)-\delta$, yield

$$E\left[P_b\right] \leqslant \frac{\varepsilon^2}{2(1-d_1-\varepsilon)} \tag{18a}$$

for all sufficiently large N, where this and all subsequent explicit expectations are over the codes in the randomly selected ensemble, where we restrict ε henceforth to satisfy $\varepsilon < 1 - d_1$, and where δ is a parameter satisfying $0 < \delta \leqslant \varepsilon$ that we specify below. Moreover, over this ensemble of codes, we see that

$$\begin{aligned} E[\Delta] &= E\left[H(U^K|Z^N)/K\right] \\ &= E\left[H(U^K)/K - I(U^K;Z^N)/K\right] \\ &\geqslant 1 - E\left[I(X^N;Z^N)/K\right] = \\ &\geqslant 1 - I_o(X;Z)/R \\ &> 1 - I_o(X;Z)/I_o(X;Y) - \varepsilon/2 = d_1 - \varepsilon/2, \end{aligned} \tag{18b}$$

where the last inequality requires us to specify δ appropriately small. The fact that $P_b \geqslant 0$, together with (18a), implies

$$P\left[P_b \geqslant \varepsilon\right] < \frac{\varepsilon/2}{1-d_1-\varepsilon}\ . \tag{19a}$$

Similarly, the fact that $\Delta \leqslant 1$, together with (18b), implies

$$P\left[\Delta \leqslant d - \varepsilon\right] \leqslant \frac{1-\varepsilon/2}{1-d_1-\varepsilon}\ . \tag{19b}$$

But (19a) and (19b) now imply that for at least one code in the ensemble both $P_b < \varepsilon$ and $\Delta > d_1-\varepsilon$. Since the code rate is $R = r_1 - \delta \geqslant r_1-\varepsilon$, it follows that (r_1,d_1) is achievable.

It is only to achieve the pair (r_2,d_2) of (17b) that we must resort to the non-obvious (and extremely clever!) coding scheme introduced by Wyner[1]. In this scheme, the "encoder" of Fig.2 now contains both an internal BSS as well as an internal encoder. The real BSS output U^K together with the internal BSS output V^L form the input to the internal encoder. We choose $R = K/N = r_2 - \delta$ (where again δ satisfies $0 < \delta \leqslant \varepsilon$ and will be specified later), but we also choose $R^* = L/N = I_o(X;Z) - \delta$. We then consider the same random coding scheme as described above except that the code rate is now

$R + R^* = I_o(X;Y) - 2\delta$. Thus, for sufficiently large N, the legitimate receiver will decode both sources, and in particular the real BSS, with

$$E[P_b] \leq \varepsilon/2. \tag{20}$$

The key part of the argument is now to note that if U^K were available to the wire-tapper, then only 2^L of the 2^{K+L} codewords would remain as possibilities so that the coding scheme would then appear to him to be exactly a random coding scheme of rate $R^* = I_o(X;Z)$, and hence he could (whether he wishes to or not!) decode V^L with average bit error probability at most δ for N sufficiently large. But Fano's inequality thus requires

$$E[H(V^L|U^K Z^N)]/L \leq h(\delta) \tag{21}$$

which as we shall soon see, dooms the wiretapper to complete ignorance about U^K. To show this, we note that

$$\begin{aligned} E[I(U^K V^L;Z^N)] &= E[H(U^K V^L) - H(U^K V^L|Z^N)] \\ &= K + L - E[H(U^K|Z^N)] - E[H(V^L|U^K Z^N)] \end{aligned} \tag{22}$$

and also that

$$E[I(U^K V^L;Z^N)] \leq E[I(X^N;Z^N)] \leq N\, I_o(X;Z). \tag{23}$$

Combining (22) and (23) and noting that $\Delta = H(U^K|Z^N)/K$ gives

$$E[\Delta] \geq 1 + L/K - I_o(X;Z)/R - E[H(V^L|U^K Z^N)]/K,$$

which with the aid of (21) and the fact that $L/K = R^*/R = [I_o(X;Z) - \delta]/R$ now gives

$$E[\Delta] \geq 1 - \delta/R - h(\delta)R^*/R.$$

Thus, for sufficiently small δ, we have

$$E[\Delta] > 1 - \varepsilon/2. \tag{24}$$

The fact that $P_b \geq 0$, together with (20), implies that $P[P_b \geq \varepsilon] \leq 1/2$. The fact that $\Delta \leq 1$, together with (24), implies that $P(\Delta \geq 1 - \varepsilon) < 1/2$. Thus, at least one code in the ensemble must have both $P_b < \varepsilon$ and $\Delta > 1 - \varepsilon = d_2 - \varepsilon$. This code has rate $R = r_2 - \delta \geq r_2 - \varepsilon$ so it follows that (r_2, d_2) is achievable.

REFERENCES

[1] A.D.Wyner, "The Wire-Tap Channel", Bell Sys.Tech.J., vol. 54, pp.1355-1387, Oct. 1975.

[2] T.M.Cover, "Broadcast Channels", IEEE Trans.Info.Th., vol. IT-18, pp.2-14, Jan. 1972.

[3] R.G.Gallager, Information Theory and Reliable Communication. New York: Wiley, 1968.

[4] S.K.Leung-Yan-Cheong, "On a Special Class of Wiretap Channels", IEEE Trans.Info.Th., vol. IT-23, pp.625-627, Sept. 1977.

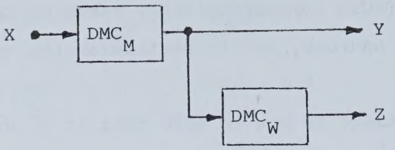

X
DMC_M
Y
DMC_W
Z

Fig.1: Wyner's Wire-Tap Channel (WTC).

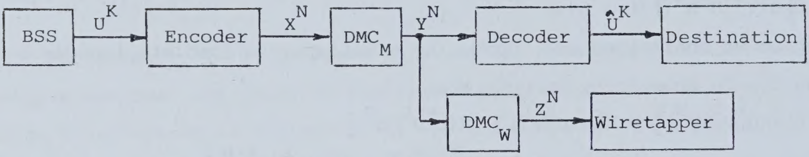
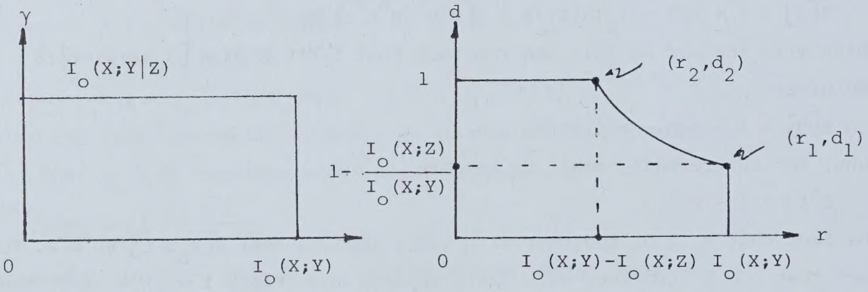

BSS
U^K
Encoder
X^N
DMC_M
Y^N
Decoder
Û^K
Destination
DMC_W
Z^N
Wiretapper

Fig.2: Coding Situation for the WTC.



(a) (b)

Fig.3: The Subregion (a) of $\mathcal{W}$ and the Subregion (b) of $\mathcal{C}$ Determined by $p_o(x)$.

ERROR BOUNDS FOR ANTI-JAMMING CODES

THOMAS ERICSON
Linköping University
Dept. of Electrical Engineering
S-581 83 Linköping, Sweden

ABSTRACT

Anti-jamming codes are codes designed to be resistant against deliberate disturbance generated by a hostile enemy (so called jamming). A characteristic feature is that the transmitted codeword depends not only on the message, but also on an additional parameter - the key - which must be delivered to the encoder and the decoder, but must be kept unknown to the jammer. In the typical jamming situation information cannot be transmitted reliably without the use of a key, and in order to achieve an exponential decrease in codeword error probability as a function of codeword length a positive key-rate must be maintained.

In the talk the above concepts will be discussed in some detail. In particular the error exponent for anti-jamming codes will be charcterized as a function of information rate and key rate.

1. INTRODUCTION

The jamming situation is characterized by the fact that the transmission channel is affected by an hostile enemy (jammer) who acts with the outspoken purpose of making the transmitted message as unreliable as possible. In general it must be assumed that the enemy knows the code as well as the source and channel statistics. To obtain protection against such a well-informed jammer it is usually necessary to use a so called anti-jamming code.

By this we understand a set of conventional codes indexed by a parameter which we call the key. For each message the legal user specifies a particular code from the given set of possible codes by the choice of the key. A typical example is the choice of transmitting frequency in a frequency hopping system. An essential assumption is that the key can be kept unknown to the jammer. It should be clear that the size of the key set (i.e. the set of possible keys) is an important system parameter. We characterize it with a parameter called key rate. The main subject of our investigation is to determine how the error probability depends on this parameter.

2. FORMAL STATEMENT OF THE PROBLEM

The mathematical model used to describe the jamming situation is the arbitrarily varying channel (AVC). It is defined as follows. Let $\mathcal{W}$ be a set of probability matrices $V: \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are finite sets, referred to as the input and output alphabet. For convenience we assume that $\mathcal{W}$ is indexed by a set $\mathcal{S}$ (which needs not to be finite) so that to each $s \in \mathcal{S}$ there corresponds a channel $V \in \overline{\mathcal{W}}$. The probability of output $y \in \mathcal{Y}$ given $x \in \mathcal{X}$ and $s \in \mathcal{S}$ is denoted $W(y|x,s)$. The matrix $W: \mathcal{X} \times \mathcal{S} \to \mathcal{Y}$ will be called the kernel. For each $n = 1,2,\ldots$ let $\mathcal{V}_n$ be the set of channels $V^n: \mathcal{X}^n \to \mathcal{Y}^n$

such that

$$V^n(\underline{y}|\underline{x}) = \prod_{i=1}^{n} W(y_i|x_i,s_i)$$

$$\underline{x} = x_1\, x_2 \dots\, x_n \in X^n$$

$$\underline{y} = y_1\, y_2 \dots\, y_n \in \mathcal{Y}^n$$

for some $\underline{s} = s_1,s_2 \dots\, s_n \in S^n$. The sequence $\{\mathcal{V}_n\}_{n=1}^{\infty}$ is the arbitrarily varying channel corresponding to the kernel W. The idea is that the jammer can generate anyone of the channels $V^n \in \mathcal{V}_n$ by a proper choice of the jamming sequence $\underline{s} = s_1,s_2 \dots\, s_n \in S^n$. The coding scheme must be chosen so as to provide reliable transmission for all $\underline{s} \in S^n$. For this purpose the concept of anti-jamming codes is needed.

Let $\mathcal{M}_z = \{1,2,\dots, M_z\}$ be an arbitrary set ($M_z = \infty$ allowed), to be called the key set. For each $z \in \mathcal{M}_z$ let (f_z,φ_z) be a pair of mappings, $f_z\colon \mathcal{M}_x \to X^n$; $\varphi_z\colon \mathcal{Y}^n \to \mathcal{M}_x$, where $\mathcal{M}_x \triangleq \{1,2,\dots,M_x\}$ is the set of messages (always $M_x < \infty$). Let Z be a random variable with range $\mathcal{M}_z$ and let (F,Φ) be a randomly selected pair from the set $\{(f_z,\varphi_z)\colon z \in \mathcal{M}_z\}$ according to the convention

$$(F,\Phi) = (f_z,\varphi_z) \leftrightarrow Z = z; \quad z \in \mathcal{M}_z.$$

The pair (F,Φ) is an anti-jamming code. Alternatively, as (F,Φ) is a random pair it is often called a random code. The random variable Z is called the key. The special case $M_z = 1$ will be referred to as a deterministic code.

Let

$$e_m(z,\underline{s}) \triangleq 1 - W^n(\varphi_z^{-1}(m)|f_z(m),\underline{s})$$

be the probability of error given $m \in \mathcal{M}_x$, $z \in \mathcal{M}_z$, and $\underline{s} \in S^n$. The average probability of error of the code (F,Φ) is

$$\bar{\lambda}(F,\Phi) \triangleq \sup_{\underline{s} \in S^n} \frac{1}{M_x} \sum_{m \in \mathcal{M}_x} E\, e_m(Z_1,\underline{s}).$$

The performance of the code (F,Φ) will be characterized by this quantity along with two further parameters to be called the information rate

$$r_x(F,\Phi) \triangleq \frac{1}{n} \log M_x$$

and the key rate

$$r_z(F,\Phi) \triangleq \frac{1}{n} \log M_z$$

($M_z < \infty$). The key rate is the rate by which information must be transmitted to the encoder and the decoder. Clearly we would like this quantity to be as small as possible. For deterministic codes the key rate is zero, but unfortunately these codes are useless in most jamming situations (see section 4). Our problem is to characterize the asymptotic relation between $\bar{\lambda}(F,\Phi)$, $r_x(F,\Phi)$ and $r_z(F,\Phi)$ as the word-length n tends to infinity.

3. PREREQUISITES

The arbitrarily varying channel was introduced by Blackwell-Breiman-Thomasian [1], who determined the capacity under the assumption that the key rate is unlimited. In fact, the code ensemble $\{(f_z,\varphi_z)\}$ was taken as the set of all possible codes (more precisely: all possible encoders, with the decoder determined by the encoder). Under this assumption the capacity was shown to be

$$C \triangleq \max_{P\in \mathcal{P}(\mathcal{X})} \min_{V\in\bar{\mathcal{W}}} \mathcal{I}(P,V)$$

where $\mathcal{P}(\mathcal{X})$ is the set of all probability distributions over $\mathcal{X}$ and where $\bar{\mathcal{W}}$ is the closed convex hull of $\mathcal{W}$. The function $\mathcal{I}(P,V)$ is Shannons mutual information function,

$$\mathcal{I}(P,V) \triangleq \sum_{x,y} P(x)\, V(y|x) \log \frac{V(y|x)}{\sum_{x'} P(x')V(y|x')}$$

Unfortunately the set of all possible encoders increases superexponentially with the word length n (its size is

$$|\mathcal{X}|^{n\cdot M_x} = \exp\{n\cdot 2^{nR_x} \cdot \log|\mathcal{X}|\}),$$

which means that the key rate is unlimited. This is clearly a very unsatisfying situation, which makes the practical value of the result very questionable. However, in 1978 Ahlswede [2] showed that the same capacity is obtained with a much smaller key ensemble. Indeed, he showed that the key ensemble may increase only polynomially in n. We refer to this result as the Random code reduction lemma ([3]p.210). It follows that the capacity is independent of the key rate. Still the key rate is a parameter of great importance as it turns out that the error exponent (reliability function) is strongly dependent on it. This is the main conclusion of the present paper. However, in order to be able to state our results properly we need still a few concepts.

4. AVC WITH SYMMETRIC KERNEL

In most jamming situations it must be assumed that the jammer has more power at his disposal than has the legal user (in the reverse situation we usually have no jamming problem at all). In a sense this means that all that can be done to the channel by the legal user can also be done by the jammer. Mathematically this situation is most easily reflected in the AVC with symmetric kernel.

Definition: an AVC is said to have symmetric kernel if $\mathcal{X} = \mathcal{S}$ and $W(y|x,s) = W(y|s,x)$ for all $(x,s,y) \in \mathcal{X} \times \mathcal{S} \times \mathcal{Y}$.

For simplicity we will state our results for AVC's with symmetric kernel. It should, however, be obvious that they immediately generalize to AVC's which can be made symmetric by a proper probabilistic transformation of the $\mathcal{S}$-space, [4].

Theorem 1: if (F,Φ) is an anti-jamming code in an AVC with symmetric kernel, then

$$\bar{\lambda}(F,\Phi) \geq \frac{M_x-1}{2M_xM_z} .$$

The proof can be found in [5]. In particular this result implies that the probability of error for a deterministic code ($M_z=1$) is always greater than or equal to $\frac{1}{4}$ (because $M_x \geq 2$) (this was observed already by Blackwell-Breiman-Thomasian [1]). We conclude that non-trivial anti-jamming codes are necessary.

A number $E > 0$ is called an achievable error exponent at rates (R_x,R_z) if for every $\delta > 0$ there is, for all sufficiently large n, some code (F,Φ) with rates $r_x(F,\Phi) \geq R_x$, $r_z(F,\Phi) \leq R_z$ and such that

$$\bar{\lambda}(F,\Phi) < \exp\{-n(E-\delta)\} .$$

The largest attainable error exponent at rates (R_x,R_z) is denoted $E(R_x,R_z)$. The corresponding quantity for unlimited key rate is denoted $E_\infty(R_x)$.

Although $E_\infty(R_x)$ is unknown in general it is known that $E_\infty(R_x) > 0$ for all $R_x < C$. This result is implicit already in [1]. An explicit general lower bound was derived by Stiglitz, [6]. Our main result is the following

Theorem 2: for an AVC with symmetric kernel $E(R_x,R_z)$ is related to $E_\infty(R_x)$ by the equality

$$E(R_x,R_z) = \min\left[E_\infty(R_x),\ R_z\right].$$

A proof of this result appears in [4]. In particular it implies that also $E(R_x,R_z) > 0$ for all $R_x < C$, independently of R_z.

5. TWO EXAMPLES

The relation between $E(R_x,R_z)$ and $E_\infty(R_z)$ is illustrated in fig 1. Although $E_\infty(R_z)$ is unknown in general it can often be determined in specific cases, especially for R_x close to C. We support this claim with two examples of some practical interest.

Example 1: (Binary adder channel)

$\mathcal{X} = S = \{0,1\}$; $\mathcal{Y} = \{0,1,2\}$

$$W(y|x,s) = \begin{cases} 1 & y = x+s \\ 0 & y \neq x+s \end{cases}$$

Combining Stiglitz' lower bound with the sphere-packing bound we get

$$E(R_x,R_z) = \min[1-h(R_x),R_z]$$

for $R_z \leq \log 4/3 - 1/3$; where h is the binary entropy function,

$$h(x) \triangleq -x \log_2 x - (1-x)\log_2(1-x) \; ; \quad 0 \leq x \leq 1.$$

Example 2: (analog channel with "hard decision")

$\mathcal{X} = \mathcal{Y} = \{0,1\}; \quad \mathcal{S} = \{s \in \mathcal{R}: s^2 \leq E_s\}$

($\mathcal{R}$ is the set of real numbers)

$$W(\cdot|\cdot,s) \triangleq \begin{bmatrix} 1-\varepsilon(s) & \varepsilon(s) \\ \varepsilon(-s) & 1-\varepsilon(-s) \end{bmatrix}$$

$$\varepsilon(s) \triangleq Q\left(\frac{\sqrt{E_x}+s}{\sigma}\right)$$

$$Q(x) \triangleq \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-t^2/2}dt$$

The channel can be interpreted as a quantized analog channel as indicated in fig. 2. By similar techniques as in the previous example one easily finds

$$E(R_x,R_z) = \min[E_{sp}(R_x),R_z]$$

for $R_z \leq R_0 - R_{cr}$; where

$$E_{sp}(R_x) \triangleq q \log \frac{q}{p^*} + (1-q) \log \frac{1-q}{1-p^*}$$

$$h(q) \triangleq 1-R_x$$

$$R_0 \triangleq 1-2 \log \left[\sqrt{p^*} + \sqrt{1-p^*}\right]$$

$$R_{cr} \triangleq 1-h\left(\frac{\sqrt{p^*}}{\sqrt{p^*} + \sqrt{1-p^*}}\right)$$

$$p^* = \frac{1}{2}\varepsilon(\sqrt{E_s}) + \frac{1}{2}\varepsilon(-\sqrt{E_s})$$

(h is still the binary entropy function).

For details we refer to [4].

REFERENCES

[1] D. Blackwell, L. Breiman, and A.J. Thomasian, "The capacities of certain channel classes under random coding", Ann.Math.Stat., 31, pp. 558-567, 1960.

[2] R. Ahlswede, "Elimination of correlation in random codes for arbitrarily varying channels", Z. Wahrscheinlichkeitstheorie u. verw. Gebiete, 44, pp. 159-175, 1978.

[3] I. Csiszar and J. Körner, "Information Theory", Akademiai Kiado, Budapest and Academic Press, 1981, Ch.2, § 6, pp. 204-233.

[4] T. Ericson, "Exponential error bounds for random codes in the arbitrarily varying channel", DFVLR, West Germany, Internal report 1983. Submitted for publication.

[5] T. Ericson, "The non-cooperative binary adder channel", Linköping Inst. of Technology, Internal report LiTH-ISY-I-0540, 1982. Submitted for publication.

[6] I.G. Stiglitz, "Coding for a class of unknown channels", IEEE Trans. Inf. Th., IT-12, No. 2., pp. 189-195, 1966.

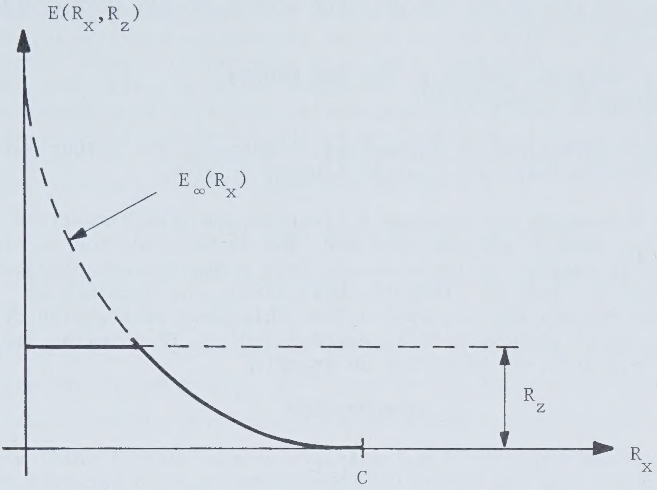
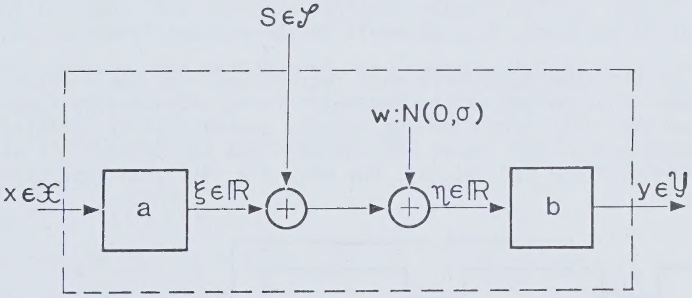


Fig. 1. The general form of the error exponent $E(R_x,R_z)$ for an AVC with symmetric kernel.


$S\in\mathcal{S}$
$w:N(0,\sigma)$
$x\in\mathcal{X}$
a
$\xi\in\mathbb{R}$
$\eta\in\mathbb{R}$
b
$y\in\mathcal{Y}$

$$a(x) = \begin{cases} \sqrt{E_x} & x=0 \\ -\sqrt{E_x} & x=1 \end{cases} \qquad b(\eta) = \begin{cases} 0 & \eta \geq 0 \\ 1 & \eta < 0 \end{cases}$$

Fig. 2. An analog channel with "hard decision".

AN ACHIEVABLE RATE REGION FOR THE MULTIPLE ACCESS CHANNEL WITH GENERALIZED FEEDBACK

FRANS M.J. WILLEMS*, EDWARD C. VAN DER MEULEN**, and J. PIETER M. SCHALKWIJK*

* Eindhoven University of Technology, Eindhoven, The Netherlands,
**Katholieke Universiteit Leuven, Leuven, Belgium.

ABSTRACT : In this paper we consider a discrete memoryless multiple access channel with two senders and one receiver. The senders and the receiver all observe different outputs of the channel. This communication situation was first considered by Carleial (IEEE-IT, Nov. 1982), who obtained an achievable rate region. Here we find a new achievable rate region by exploiting the novel technique of backward decoding. This region improves upon Carleial's region, at least for an example.

I. INTRODUCTION

The discrete memoryless (d.m.) multiple access channel (MAC) with noiseless feedback from the output of the channel to both encoders was first investigated by Gaarder and Wolf [1]. They showed that for the binary erasure MAC feedback enlarges the capacity region. Cover and Leung [2] generalized Gaarder and Wolf's two-stage transmission scheme in that they superimposed both stages on each other. With this coding method (which they named superposition block Markov encoding) they found an achievable rate region for the MAC with feedback.

Partial feedback for the MAC, i.e. noiseless feedback to only one encoder, was introduced by Dueck [3]. Carleial [4] and Willems and van der Meulen [5] showed that the region obtained by Cover and Leung [2] for the MAC with (full) feedback, is achievable in the partial feedback situation as well.

King [6] obtained achievable rate regions for d.m. MAC's where both encoders observe an output of the MAC which is not identically to the decoder's output. This situation was further generalized by Carleial [4], who considered a d.m. MAC where the encoders and the decoder all observe an other output of the MAC (Fig.1). For this d.m. MAC with generalized feedback we will give an achievable rate region in this paper. The achievability proof involves beside a three-level superposition block

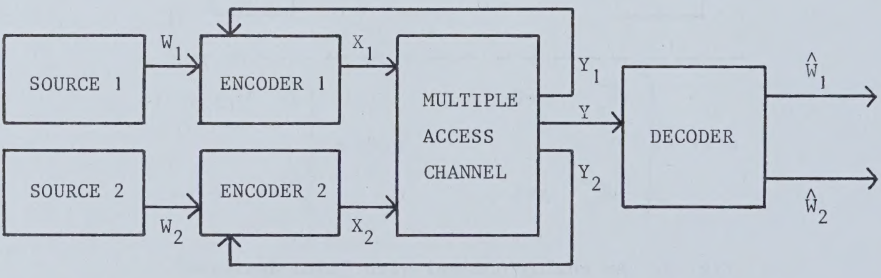


Fig. 1. The discrete memoryless multiple access channel with generalized feedback.

Markov encoding scheme the novel technique of backward decoding.

Because of the complexity of the region obtained by Carleial [4] it is difficult to compare this region with ours. However, at least for the case where the generalized feedback situation specializes to the cribbing encoders situation 2 or 5 (see Willems and van der Meulen [7]) Carleial's region is strictly inside the region we find here. It is easily checked that King's [6] region, if we set $\hat{Y}^* \equiv \phi$, is strictly inside our achievable rate region for the case of equal feedback outputs.

II. DEFINITIONS AND STATEMENT OF RESULT

A (two-user) d.m. MAC with generalized feedback outputs is denoted by $(\mathcal{X}_1 \times \mathcal{X}_2, P^*(y,y_1,y_2|x_1,x_2), \mathcal{Y} \times \mathcal{Y}_1 \times \mathcal{Y}_2)$, where $\mathcal{X}_1$ and $\mathcal{X}_2$ are the input alphabets, $\mathcal{Y}$ is the output alphabet, $\mathcal{Y}_1$ and $\mathcal{Y}_2$ are the feedback output alphabets, and $P^*(y,y_1,y_2|x_1,x_2)$ is the transition probability matrix. All alphabets are finite and for every N transmissions we have that

$$\Pr\{y^N,y_1^N,y_2^N|x_1^N,x_2^N\} = \prod_{n=1,N} P^*(y_n,y_{1n},y_{2n}|x_{1n},x_{2n}), \tag{1}$$

where $y^N \in \mathcal{Y}^N$, $y_1^N \in \mathcal{Y}_1^N$, $y_2^N \in \mathcal{Y}_2^N$, $x_1^N \in \mathcal{X}_1^N$, and $x_2^N \in \mathcal{X}_2^N$.

The message sources 1 and 2 produce random integers $W_1 \in \{1,2,\ldots,M_1\}$ and $W_2 \in \{1,2,\ldots,M_2\}$ respectively, at the beginning of each block of N channel uses. Each pair (w_1,w_2) occurs with probability $1/M_1M_2$.

Each encoder is completely described by a set of N encoding functions. These functions map the message and the sequence of corresponding feedback outputs into the next channel input. Hence, if we denote $(Y_{11},Y_{12},\ldots,Y_{1n-1})$ by Y_1^{n-1} et cetera then

$$x_{1n} = f_{1n}(W_1,Y_1^{n-1}), \tag{2a}$$

$$x_{2n} = f_{2n}(W_2,Y_2^{n-1}),\ n=1,2,\ldots,N. \tag{2b}$$

The decoder gives estimates of W_1 and W_2 based on his knowledge of the received sequence of channel outputs. That is

$$(\hat{w}_1,\hat{w}_2) = g(Y^N). \tag{3}$$

An (M_1,M_2,N,P_e)-code for the d.m. MAC with generalized feedback consists of two sets of N encoding functions and a decoding function such

$$\Pr\{(\hat{W}_1,\hat{W}_2) \neq (W_1,W_2)\} = P_e. \tag{4}$$

A rate pair (R_1,R_2) is said to be achievable for a d.m. MAC with generalized feedback, if for any $\varepsilon > 0$ there exists for all N sufficiently large an (M_1,M_2,N,P_e)-code for that MAC such that $(1/N)\log(M_1) \geq R_1 \geq 0$, $(1/N)\log(M_2) \geq R_2 \geq 0$, and $P_e \leq \varepsilon$.

$\mathcal{C}_{gf}$, the capacity region for a d.m. MAC with generalized feedback is the closure of the set of all achievable rate pairs (R_1,R_2).

We can now state the result of this paper :

Theorem : For a d.m. MAC $(\mathcal{X}_1 \times \mathcal{X}_2, P^*(y,y_1,y_2|x_1,x_2), \mathcal{Y} \times \mathcal{Y}_1 \times \mathcal{Y}_2)$ the generalized feedback capacity region $\mathcal{C}_{gf} \supseteq \mathcal{R}_{gf}$, where the achievable region $\mathcal{R}_{gf}$ is defined as follows :

$$
\begin{aligned}
\mathcal{R}_{gf} \overset{\Delta}{=} \{(R_1,R_2) : {} & R_1 = R_{12} + R_{11},\ R_2 = R_{21} + R_{22}, \\
& 0 \leq R_{12} \leq I(V_1;Y_2|X_2,U),\ 0 \leq R_{21} \leq I(V_2;Y_1|X_1,U), \\
& 0 \leq R_{11} \leq I(X_1;Y|X_2,V_1,U), \\
& 0 \leq R_{22} \leq I(X_2;Y|X_1,V_2,U), \\
& R_{11} + R_{22} \leq I(X_1,X_2;Y|V_1,V_2,U), \\
& R_{12} + R_{21} + R_{11} + R_{22} \leq I(X_1,X_2;Y) \text{ for} \\
& P(u,v_1,v_2,x_1,x_2,y,y_1,y_2) = P(u)P(v_1|u)P(v_2|u)P(x_1|v_1,u) \\
& \qquad .P(x_2|v_2,u)P^*(y,y_1,y_2|x_1,x_2)\}.
\end{aligned}
$$

Note that since the cardinalities of the auxiliary random variables are not explicitely bounded the region $\mathcal{R}_{gf}$ might not be closed. It is not difficult however to establish its convexity.

The proof of the theorem involves backward decoding. The next section is devoted to this new decoding technique.

III. BACKWARD DECODING

In order to explain the idea behind backward decoding, we first consider a coding policy which is frequently used to prove multi-user coding theorems, namely superposition block Markov encoding (SBME). This method was introduced for the MAC with feedback by Cover and Leung [2] and succesfully applied by Cover and El Gamal [8] for the relay channel. The idea behind SBME can be summarized as follows :

Transmission occurs in B blocks (see Fig.2). In block b, b=1,2,..., b=1,2,...,B-1, the transmitters send to the receiver a certain amount of "fresh information," however with a rate too high to allow reliable decoding. As there is ambiguity left for the receiver, the transmitters in the next block cooperate to send "resolution information" to the receiver. Superimposed on this resolution information, the transmitters in addition send "high rate" fresh information.

Decoding in the case of SBME happens in the following way :

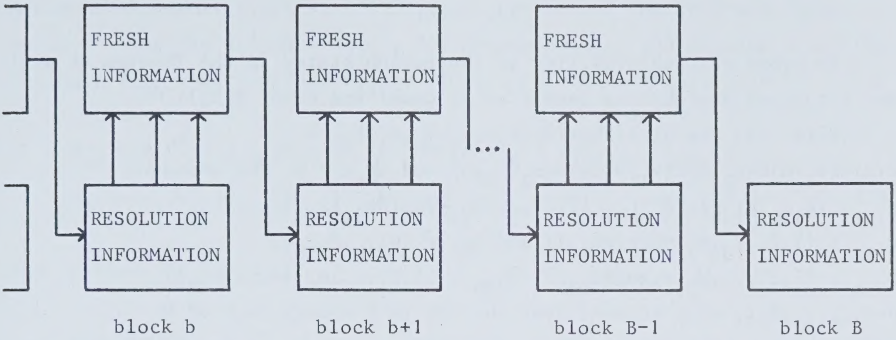


Fig. 2. Superposition block Markov encoding.

Assume that the resolution information in block b has already been decoded. In order to decode the fresh information in block b, the receiver first decodes the resolution information in block b+1. Using this information from block b+1, the fresh information in block b can be decoded succesfully, conditionally on the already decoded resolution information in block b. Methods like list coding and lexicographical indexing (Cover and Leung [2]), list coding and random partitioning (Cover and El Gamal [8]), or non-random partitioning and restricted decoding (Willems and van der Meulen [5]) have been used in literature to accomplish this.

While this decoding method proved to be optimal for the degraded relay channel (Cover and El Gamal [8]), and also for a class of d.m. MAC's with feedback and partial feedback (Willems [9]), it turns out to be suboptimal in the case of cribbing encoders (Willems and van der Meulen [7]). The reason for this suboptimality is that decoding happens essentially in a sequential way. Carleial [10, page 26] refers to this method as step by step decoding and as pointed out by Han and Kobayashi [11] this can be inferior to simultaneous decoding. In order to obtain in each block a simultaneous form of decoding, in [7] the novel idea of backward decoding is introduced. We will now describe this decoding method :

The receiver starts decoding only after all B blocks have been received (see again Fig.2). In block B only resolution information is decoded, which is used to transform the high rate fresh information in block B-1 into "low rate" information. Next the receiver decodes simultaneously in block B-1 this fresh information together with the resolution information needed for block B-2. With this resolution information the decoder transforms the high rate information in block B-2 into decodable low rate information. At this stage again simultaneous decoding can be applied. Continuing like this the receiver successively decodes decodes the fresh information in all blocks in a block-simultaneous way, finally in block 1. In general, the principle of non-random partitions and restricted decoding proposed in [5] can be used to transform high rate information into low rate information. However, in the generalized feedback case these partitions occur in a degraded form in that each cell of these partitions contains only one element.

Now that simultaneous decoding of the fresh and the resolution information is possible, higher resolution rates are achievable. This in general increases the overall transmission rates.

In the following section we will apply the described decoding method to prove the stated Theorem.

IV. PROOF

To prove the achievability of the region stated in the Theorem we will use ε-typical N-sequences (see f.e. El Gamal and Cover [12]).

First fix the distribution $P(u,v_1,v_2,x_1,x_2) = P(u)P(v_1|u)P(v_2|u)P(x_1|v_1,u)P(x_2|v_2,u)$ and an $\varepsilon > 0$. The messages $W_{1b} = (W_{12b},W_{11b})$, $W_{12b} \in \{1,2,\ldots,M_{12}=\exp(NR_{12})\}$, $W_{11b} \in \{1,2,\ldots,M_{11}=\exp(NR_{11})\}$ and $W_{2b} = (W_{21b},W_{22b})$, $W_{21b} \in \{1,2,\ldots,M_{21}=\exp(NR_{21})\}$, $W_{22b} \in \{1,2,\ldots,M_{22}=\exp(NR_{22})\}$, $b=1,2,\ldots,B-1$, will be sent over the MAC in B blocks each of N transmissions. The messages W_{12b}, W_{11b}, W_{21b}, and W_{22b}, $b=1,2,\ldots,B-1$, are uniformly distributed and independent of each other. It should be noted that if $B \to \infty$, the overall rates $R_{12}(B-1)/B \to R_{12}$, $R_{11}(B-1)/B \to R_{11}$, $R_{21}(B-1)/B \to R_{21}$, and $R_{22}(B-1)/B \to R_{22}$. We will use a random coding argument to demonstrate the achievability of the region $\mathcal{R}_{gf}$. In each of the B blocks we will use the same codebook. This is constructed in the following way :

(a) Generate $M_{12}M_{21}$ sequences $\underline{u} = (u_1,u_2,\ldots,u_N)$ each with probability $\Pr\{\underline{u}\} = \Pi_{n=1}^{N} P(u_n)$. Label them $\underline{u}(w_0)$, $w_0 = (w_{01},w_{02})$, $w_{01} \in \{1,2,\ldots,M_{12}\}$, $w_{02} \in \{1,2,\ldots,M_{21}\}$.

(b) For every $\underline{u}(w_0)$ generate M_{12} sequences $\underline{v}_1 = (v_{11},v_{12},\ldots,v_{1N})$ each with probability $\Pr\{\underline{v}_1|\underline{u}(w_0)\} = \Pi_{n=1}^{N} P(v_{1n}|u_n(w_0))$. Label them $v_1(w_0,w_{12})$, $w_{12} \in \{1,2,\ldots,M_{12}\}$.

(c) For every $\underline{u}(w_0)$ also generate M_{21} sequences $\underline{v}_2 = (v_{21},v_{22},\ldots,v_{2N})$ each with probability $\Pr\{\underline{v}_2|\underline{u}(w_0)\} = \Pi_{n=1}^{N} P(v_{2n}|u_n(w_0))$. Label them $v_2(w_0,w_{21})$, $w_{21} \in \{1,2,\ldots,M_{21}\}$.

(d) For every pair $\underline{u}(w_0),\underline{v}_1(w_0,w_{12})$ generate M_{11} sequences $\underline{x}_1 = (x_{11},x_{12},\ldots,x_{1N})$ each with probability $\Pr\{\underline{x}_1|\underline{v}_1(w_0,w_{12}),\underline{u}(w_0)\} = \Pi_{n=1}^{N} P(x_{1n}|v_{1n}(w_0,w_{12}),u_n(w_0))$. Label them $\underline{x}_1(w_0,w_{12},w_{11})$, $w_{11} \in \{1,2,\ldots,M_{11}\}$.

(e) Finally for every pair $\underline{u}(w_0),\underline{v}_2(w_0,w_{21})$ generate M_{22} sequences $\underline{x}_2 = (x_{21},x_{22},\ldots,x_{2N})$ each with probability $\Pr\{\underline{x}_2|\underline{v}_2(w_0,w_{21}),\underline{u}(w_0)\} = \Pi_{n=1}^{N} P(x_{2n}|v_{2n}(w_0,w_{21}),u_n(w_0))$. Label them $\underline{x}_2(w_0,w_{21},w_{22})$, $w_{22} \in \{1,2,\ldots,M_{22}\}$.

The messages w_{12b}, w_{11b}, w_{21b}, and w_{22b}, $b=1,2,\ldots,B-1$ are now encoded in the following way (superposition block Markov encoding) : In block 1

$$\underline{x}_{11} = \underline{x}_1((1,1),w_{121},w_{111}),$$
$$\underline{x}_{21} = \underline{x}_2((1,1),w_{211},w_{221}). \tag{5a}$$

Assume that before block b, b=2,3,...,B, because of the feedback links, encoder 1 has the estimate $\hat{w}'_{21b-1}$ for message w_{21b-1} and encoder 2 has the estimate $\hat{w}''_{12b-1}$ for message w_{12b-1}. Then in block b, b=2,3,...,B-1

$$\left.\begin{array}{l} \underline{x}_{1b} = \underline{x}_1(\hat{w}'_{0b}, w_{12b}, w_{11b}) \\ \underline{x}_{2b} = \underline{x}_2(\hat{w}''_{0b}, w_{21b}, w_{22b}) \end{array}\right\} \quad \begin{array}{l} \text{where } \hat{w}'_{0b} = (w_{12b-1}, \hat{w}'_{21b-1}) \\ \text{and } \hat{w}''_{0b} = (\hat{w}''_{12b-1}, w_{21b-1}), \end{array} \tag{5b}$$

and in block B

$$\left.\begin{array}{l} \underline{x}_{1B} = \underline{x}_1(\hat{w}'_{0B}, 1, 1) \\ \underline{x}_{2B} = \underline{x}_2(\hat{w}''_{0B}, 1, 1) \end{array}\right\} \quad \begin{array}{l} \text{where } \hat{w}'_{0B} = (w_{12B-1}, \hat{w}'_{21B-1}) \\ \text{and } \hat{w}''_{0B} = (\hat{w}''_{12B-1}, w_{21B-1}). \end{array} \tag{5c}$$

We will now describe the decoding process in which the principle of backward decoding is used to find the transmitted information. First in the last block, block B, the decoder looks for $\hat{w}_{0B} = (\hat{w}_{01B}, \hat{w}_{02B})$ such that

$$\begin{array}{l} (\underline{u}(\hat{w}_{0B}), \underline{v}_1(\hat{w}_{0B}, 1), \underline{v}_2(\hat{w}_{0B}, 1), \underline{x}_1(\hat{w}_{0B}, 1, 1), \underline{x}_2(\hat{w}_{0B}, 1, 1), \underline{y}_B) \\ \quad \in \mathcal{A}_\varepsilon(U, V_1, V_2, X_1, X_2, Y). \end{array} \tag{6a}$$

Now in block B-1 the decoder already has an estimate for w_{12B-1} and w_{21B-1} namely $\hat{w}_{12B-1} = \hat{w}_{01B}$ and $\hat{w}_{21B-1} = \hat{w}_{02B}$. Then he looks for the triple $(\hat{w}_{0B-1}, \hat{w}_{11B-1}, \hat{w}_{22B-1})$ such that

$$\begin{array}{l} (\underline{u}(\hat{w}_{0B-1}), \underline{v}_1(\hat{w}_{0B-1}, \hat{w}_{12B-1}), \underline{v}_2(\hat{w}_{0B-1}, \hat{w}_{21B-1}), \\ \underline{x}_1(\hat{w}_{0B-1}, \hat{w}_{12B-1}, \hat{w}_{11B-1}), \underline{x}_2(\hat{w}_{0B-1}, \hat{w}_{21B-1}, \hat{w}_{22B-1}), \underline{y}_{B-1}) \\ \quad \in \mathcal{A}_\varepsilon(U, V_1, V_2, X_1, X_2, Y). \end{array} \tag{6b}$$

With the output $\hat{w}_{0B-1} = (\hat{w}_{01B-1}, \hat{w}_{02B-1})$ of this decoding step the decoder forms $\hat{w}_{12B-2} = \hat{w}_{01B-1}$ and $\hat{w}_{21B-2} = \hat{w}_{02B-1}$. Continuing in this way the decoder successively finds $\hat{w}_{0B}$, $(\hat{w}_{0B-1}, \hat{w}_{12B-1}, \hat{w}_{21B-1}, \hat{w}_{11B-1}, \hat{w}_{22B-1}), \ldots,$ $(\hat{w}_{0b+1}, \hat{w}_{12b+1}, \hat{w}_{21b+1}, \hat{w}_{11b+1}, \hat{w}_{22b+1})$, b=1,2,...,B-2. Now in order to find the information in block b, the decoder first sets

$$\begin{array}{l} \hat{w}_{12b} = \hat{w}_{01b+1}, \\ \hat{w}_{21b} = \hat{w}_{02b+1}. \end{array} \tag{6c}$$

Then in block b the decoder looks for $(\hat{w}_{0b}, \hat{w}_{11b}, \hat{w}_{22b})$ such that

$$(\underline{u}(\hat{w}_{0b}),\underline{v}_1(\hat{w}_{0b},\hat{w}_{12b}),\underline{v}_2(\hat{w}_{0b},\hat{w}_{21b}),\underline{x}_1(\hat{w}_{0b},\hat{w}_{12b},\hat{w}_{11b}),$$
$$\underline{x}_2(\hat{w}_{0b},\hat{w}_{21b},\hat{w}_{22b}),\underline{y}_b) \in \mathcal{A}_\varepsilon(U,V_1,V_2,X_1,X_2,Y). \tag{6d}$$

We remark here that in block 1 this operation is simpler since $w_{01} = (1,1)$. It should be noted that (6b) and (6d) are *restricted decoding* steps since the estimates for the messages w_{12} and w_{21} are already prescribed by the outcomes of a preceding decoding step.

In order to obtain *cooperation*, after each block b, b=1,2,...,B-1, encoder 1 chooses $\hat{w}'_{21b}$ such that

$$(\underline{u}(\hat{w}'_{0b}),\underline{v}_1(\hat{w}'_{0b},w_{12b}),\underline{v}_2(\hat{w}'_{0b},\hat{w}'_{21b}),\underline{x}_1(\hat{w}'_{0b},w_{12b},w_{11b}),\underline{y}_{1b})$$
$$\in \mathcal{A}_\varepsilon(U,V_1,V_2,X_1,Y_1), \tag{7a}$$

and encoder 2 chooses $\hat{w}''_{12b}$ such that

$$(\underline{u}(\hat{w}''_{0b}),\underline{v}_1(\hat{w}''_{0b},\hat{w}''_{12b}),\underline{v}_2(\hat{w}''_{0b},w_{21b}),\underline{x}_2(\hat{w}''_{0b},w_{21b},w_{22b}),\underline{y}_{2b})$$
$$\in \mathcal{A}_\varepsilon(U,V_1,V_2,X_2,Y_2). \tag{7b}$$

Here $\hat{w}'_{0b} = (w_{12b-1},\hat{w}'_{21b-1})$ and $\hat{w}''_{0b} = (\hat{w}''_{12b-1},w_{21b-1})$ are formed from the results of decoding operations performed by the encoders after block b-1. Note that the encoders can form $\hat{w}''_{12b}$ and $\hat{w}'_{21b}$ only since they have access to the feedback outputs. Furthermore $\hat{w}'_{01} = \hat{w}''_{01} = (1,1)$ by (5a).

From (5) to (7) we see that w_{11} and w_{22} contain information which is directly sent to the decoder, while w_{12} and w_{21} contain high rate fresh information which is sent to respectively encoder 2 and encoder 1 via the feedback outputs of the MAC. Both encoders then cooperate in sending resolution information by means of the message w_0 to the decoder.

If in a decoding step there are no message indices to satisfy the decoding rule or the are more than one (vector), an index (vector) is choosen at random.

Now what is left to prove is that the average *error probability* averaged over the ensemble of codes for B-1 message quartets can be made smaller than any positive number if (R_1,R_2) is inside $\mathcal{R}_{gf}$. It is easily seen that the constraints $R_{11} < I(X_1;Y|X_2,V_1,U)$, $R_{22} < I(X_2;Y|X_1,V_2,U)$, $R_{11} + R_{22} < I(X_1,X_2;Y|V_1,V_2,U)$, and $R_{12} + R_{21} + R_{11} + R_{22} < I(X_1,X_2;Y)$ are necessary to obtain a small error probability for the decoding operations performed by the decoder ((6a), (6b), and (6d)). It is also clear that the encoders can decode the information sent by the opposite encoder (steps (7a) and (7b)) with small error probability if $R_{12} < I(V_1;Y_2|X_2,U)$ and $R_{21} < I(V_2;Y_1|X_1,U)$. For more details concerning this last part of the achievability proof we refer to Willems [13].

V. CONCLUSIONS

The theorem in this paper is a very general result in that a number of achievability results in multi-user literature are specializations of it. We will mention these specializations shortly :

(a) If we set in $\mathcal{R}_{gf}$ $Y_1 \equiv Y_2 \equiv \phi$ (=constant) and $U \equiv V_1 \equiv V_2 \equiv \phi$ it follows that the classical capacity region for the d.m. MAC (found by Ahlswede [14]) is achievable.

(b) If we set $Y_1 \equiv \phi$, $Y_2 \equiv Y$, $V_1 \equiv X_1$, and $V_2 \equiv \phi$ it follows that in the partial feedback MAC situation the Cover-Leung [2] region is achievable.

(c1) If we set $Y_1 \equiv \phi$, $Y_2 \equiv X_1$, $V_1 \equiv X_1$, $V_2 \equiv \phi$ the achievability part for the coding theorem for MAC cribbing encoders situation 2 follows.

(c2) If we set $Y_1 \equiv X_2$, $Y_2 \equiv X_1$, $V_1 \equiv X_1$, and $V_2 \equiv X_2$ the achievablity part for the coding theorem for MAC cribbing encoders situation 5 follows.

(c3) If we set $Y_1 \equiv X_2$, $Y_2 \equiv X_1$, $V_1 \equiv X_1$, $V_2 \equiv T$, and $X_2 \equiv T$ the achievability part for the coding theorem for MAC cribbing encoders situation 6 follows (see Willems and van der Meulen [7]).

(d) If we set $Y_1 \equiv \phi$, $Y_2 \equiv Y_1$, $V_1 \equiv Q$, $V_2 \equiv \phi$, and $U \equiv X_2$ it follows that the rate R_0 as defined by El Gamal and Aref [15] is achievable for any d.m. relay channel.

From this last result we can obtain the achievability parts for the (semi-) deterministic relay channel (El Gamal and Aref [15]) and the (reversely) degraded relay channel (Cover and El Gamal [8]).

It is interesting to see that, if we apply backward decoding, our original three-level superposition encoding scheme is reduced, as far as decoding is concerned, to a two-level Slepian-Wolf [16] scheme. The common message, which contains resolution information about w_{12} and w_{21} in the previous block is transmitted via the triple $(\underline{u},\underline{v}_1,\underline{v}_2)$ while the private messages w_{11} and w_{22} are transmitted directly to the decoder via $\underline{x}_1$ and $\underline{x}_2$.

King [6] applies a technique which was introduced by Cover and El Gamal [8] for the relay channel. In King's thesis, similar to Cover and El Gamal, a hypothetical output $\hat{Y}^*$ is constructed by the encoders and sent to the decoder. The decoder can use this output $\hat{Y}^*$ together with the normal output Y to reconstruct the transmitted messages. Our achievability scheme could also be extended in this way. The encoders in our situation could also construct one or more hypothetical outputs and send them to the decoder. Still the block Markov superposition encoding scheme combined with backward decoding can be used then.

REFERENCES

[1] N.T. Gaarder and J.K. Wolf, "The capacity region of a multiple-access discrete memoryless channel can increase with feedback," IEEE Trans. Inform. Theory, vol. IT-21, pp. 100-102, Jan. 1975.

[2] T.M. Cover and C.S.K. Leung, "An achievable rate region for the multiple-access channel with feedback," IEEE Trans. Inform. Theory, vol. IT-27, pp. 292-298, May 1981.

[3] G. Dueck, "Semi-feedback for multi-user channels," manuscript, Universität Bielefeld, April 1979.

[4] A.B. Carleial, "Multiple-access channels with different generalized feedback signals," IEEE Trans. Inform. Theory, vol. IT-28, pp.841-850, Nov.1982.

[5] F.M.J. Willems and E.C. van der Meulen, "Partial feedback for the discrete memoryless multiple access channel," IEEE Trans. Inform. Theory, vol. IT-29, pp. 287-290, March 1983.

[6] R.G. King, "Multiple access channels with generalized feedback," Ph. D. dissertation, Stanford University, Stanford, March 1978.

[7] F.M.J. Willems and E.C. van der Meulen, "The discrete memoryless multiple access channel with cribbing encoders," submitted to IEEE Trans. Inform. Theory.

[8] T.M. Cover and A. El Gamal, "Capacity theorems for the relay channel," IEEE Trans. Inform. Theory, vol. IT-25, pp. 572-584, Sept. 1979.

[9] F.M.J. Willems, "The feedback capacity region of a class of discrete memoryless multiple access channels," IEEE Trans. Inform. Theory, vol. IT-28, pp. 93-95, Jan. 1982.

[10] A.B. Carleial, "On the capacity of multiple-terminal communication networks," Technical Report No. 6603-1, Information Systems Laboratory, Center for Systems Research, Stanford University, Stanford, August 1975.

[11] T.S. Han and K. Kobayashi, "A new achievable rate region for the interference channel," IEEE Trans. Inform. Theory, vol. IT-27, pp. 49-60, Jan. 1981.

[12] A. El Gamal and T.M. Cover, "Multiple-user information theory," Proc. of the IEEE, vol. 68, pp. 1466-1483, Dec. 1980.

[13] F.M.J. Willems, "Informationtheoretical results for the discrete memoryless multiple access channel," Ph.D. dissertation, Katholieke Universiteit Leuven, Leuven, Nov. 1982.

[14] R. Ahlswede, "Multi-way communication channels," in Proc. 2nd Int. Symp. Information Theory, Tsahkadsor, Armenian S.S.R., 1971, pp. 23-52. Publishing House of the Hungarian Academy of Sciences, 1973.

[15] A. El Gamal and M. Aref, "The capacity of the semi-deterministic relay channel," IEEE Trans. Inform. Theory, vol. IT-28, p. 536, May 1982.

[16] D. Slepian and J.K. Wolf, "A coding theorem for multiple access channels with correlated sources," Bell Syst. Tech. J., vol. 52, pp. 1037-1076, Sept. 1973.

THE BINARY AND GAUSSIAN RATE-DISTORTION FUNCTIONS FOR A DEGRADED SOURCE NETWORK

Toby Berger and Chris Heegard
School of Electrical Engineering
Cornell University
Ithaca, NY 14853

ABSTRACT

We consider the problem of encoding a discrete memoryless source, with a fidelity criterion, when correlated side information may or may not be available to the decoder. It is assumed that the side information is not available to the encoder. We determine the rate-distortion function $R(D_1,D_2)$, where D_1 is the distortion achieved with side information and D_2 is the distortion achieved without it. We also generalize to the case of m decoders, each of which is privy to its own side information. An appropriately defined $\underline{D}$ -admissible rate for this general case is shown to equal $R(\underline{D})$ when the side information sources satisfy a specified degradedness condition. Explicit results are obtained in the quadratic Gaussian and binary Hamming cases.

1. INTRODUCTION

In a celebrated paper Wyner and Ziv [1] extended rate-distortion theory to the case in which side information is present at the decoder. They knew from Slepian and Wolf's earlier treatment of distortionless coding of correlated information sources [2] that each cell of the encoder's partition of the set of typical source sequences should consist of widely dispersed elements.

Suppose the side information were to fail to reach the decoder in the Wyner-Ziv problem. Then the decoder would know only the partition cell index. Because the elements of said cell are so widely dispersed, this knowledge would be virtually useless for reproducing the source output. This contingency has led us to consider the rate-distortion problem for cases in which it is not known whether or not side information will be present. An equivalent formulation of this problem has two decoders, one of which receives the side information and the other of which does not (cf. Figure 1).

We determine the rate-distortion function $R(D_1,D_2)$, where D_1 is the distortion achieved with the side information and D_2 is the distortion achieved without it. This is accomplished by proving both a source coding theorem and its converse. We also generalize to cases in which there are m decoders each of which is privy to its own side information. An appropriately defined $\underline{D}$-admissible rate for this general case is shown to equal $R(\underline{D})$ when the side information sources satisfy a specified degradedness condition.

Explicit results are provided in the quadratic Gaussian case and in the binary symmetric Hamming case.

2. PROBLEM STATEMENT AND THEOREM STATEMENT

Let $(\mathcal{X},\mathcal{Y},p(x,y))$ be a discrete memoryless 2-source with generic random variables X and Y. For $i \in \{1,2\}$, let $\hat{\mathcal{X}}_i$ be a reconstruction alphabet and let

---

This work was supported in part by NSF Grants ENG79-21976 and ECS-8204886 and DOE Contract DEAC-0180RA50419.

$$d_i : \mathcal{X} \times \hat{\mathcal{X}}_i \to [0,\infty)$$

be a distortion measure. An (n,M,D_1,D_2) code consists of an encoder

$$f : \mathcal{X}^n \to \{0,1,\ \cdot\ \cdot\ \cdot\ ,M-1\}$$

and two decoders

$$g_1 : \{0,1,\ \cdot\ \cdot\ \cdot\ ,M-1\} \times \mathcal{Y}^n \to \hat{\mathcal{X}}_1^n$$

$$g_2 : \{0,1,\ \cdot\ \cdot\ \cdot\ ,M-1\} \to \hat{\mathcal{X}}_2^n.$$

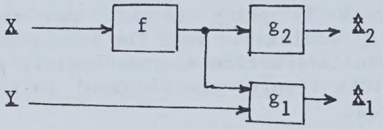


Figure 1. Block diagram of the coding system

The expected distortion $\underline{D} = (D_1,D_2)$ for the code is given by

$$D_i = Ed_i(\underline{X},\hat{\underline{X}}_i) \triangleq E\,\frac{1}{n}\sum_{k=1}^{n} d_i(X_k,\hat{X}_{ik}),\ (i = 1,2)$$

where

$$\hat{\underline{X}}_1 = g_1(f(\underline{X}),\underline{Y}) \text{ and } \hat{\underline{X}}_2 = g_2(f(\underline{X}))$$

The rate R is said to be $\underline{D}$-admissible if for every $\epsilon > 0$ there exists for some n an $(n,M,D_1 + \epsilon,D_2 + \epsilon)$ code with $n^{-1}\log M \le R + \epsilon$. We seek to determine the rate-distortion function $R(D_1,D_2) = R(\underline{D})$ defined by

$$R(\underline{D}) = \inf\{R : R \text{ is } \underline{D}\text{-admissible}\}$$

Define

$$R_0(\underline{D}) \triangleq \min_{P(\underline{D})} I(X;W) + I(X;U|Y,W)$$

where $P(\underline{D})$ is the set of all discrete random variables (W,U) jointly distributed with the generic random variables (X,Y), such that the following conditions are satisfied:

(i) Y -o- X -o- (W,U) is a Markov string

(ii) there exist functions $\hat{X}_1(W,U,Y)$ and $\hat{X}_2(W)$ such that $Ed_i(X,\hat{X}_i) \le D_i, (i=1,2)$

Theorem 1. $R(\underline{D}) = R_0(\underline{D})$

Proof. The proof of Theorem 1 is divided into two parts. The proof that $R_0(\underline{D}) \ge R(\underline{D})$, i.e. that $R_0(\underline{D})$ is $\underline{D}$-admissible, is based on a more-or-less standard [3] random coding argument that is too lengthy to present here. The converse source coding theorem, $R_0(\underline{D}) \le R(\underline{D})$, is proven in Section 3.

Remark: The informed decoder g_1 is faced with a Wyner-Ziv problem, while the uninformed decoder g_2 is faced with an ordinary rate-distortion problem. Therefore, our prescription for calculating $R_0(\underline{D})$ must reduce accordingly in the appropriate special cases. In this regard it is not difficult to show that, if only the informed decoder were present, then no loss of generality would result from setting $W = \text{const}$. This reduces the task of calculating $R_0(\underline{D})$ to minimization of $I(X;U|Y)$ over all U such that Y -o- X -o- U and there exists $\hat{X}(U,Y)$ satisfying $Ed_1(X,\hat{X}) \le D_1$. This is the prescription [1] for the Wyner-Ziv rate-distortion function, $R_{W-Z}(D_1)$.

3. THE CONVERSE

To prove that $R(\underline{D}) \ge R_0(\underline{D})$ we first show that $R_0(\underline{D})$ is convex. Let

$$\bar{R}_0(\underline{D}) = \min_{\bar{P}(\underline{D})} I(X;W|T) + I(X;U|Y,W,T)$$

where $\bar{P}(\underline{D})$ is the set of all random variables (W,U,T) jointly distributed with the generic source variables (X,Y) such that

(i) Y -o- X -o- W,U,T is a Markov string
(ii) $\exists\ \hat{X}_1(W,U,T,Y)$ and $\hat{X}_2(W,T)$ such that $Ed_i(X,\hat{X}_i) \le D_i, i \in \{1,2\}$
(iii) $\underline{T}$ is independent of X

Note that $R_0(\underline{D})$ is the lower convex envelope of $R_0(\underline{D})$. Since $I(X;T) = 0$ we have $I(X;W|T) = I(X;W,T)$. Upon defining a new random variable $W' = (W,T)$, we see that $R_0(\underline{D}) \ge R_0(\underline{D})$. Thus $R_0(\underline{D})$ lies on or below its lower convex envelope, so $R_0(\underline{D})$ must be convex.

We now shall show that, if an (n,M,D_1,D_2) code exists, then

$$n^{-1}\log M \ge I(X;W) + I(X;U|Y,W)$$

for random variables $(W,U) \in P(\underline{D})$. If we define $J = f(\underline{X})$, then

$$nR \triangleq \log M \ge H(J) \ge I(\underline{X};J)$$

$$= I(\underline{X};J,\underline{Y}) - I(\underline{X};\underline{Y}|J)$$

$$= \sum_{k=1}^{n} I(X_k;J,\underline{Y}|\underline{X}_k^-) - I(\underline{X};Y_k|J,\underline{Y}_k^-)$$

where $\underline{X}_1^- \triangleq \emptyset$ and $\underline{X}_k^- \triangleq (X_1,X_2, \dots, X_{k-1})$ for $k > 1$; similarly, $\underline{Y}_n^+ \triangleq \emptyset$ and $\underline{Y}_k^+ = (Y_{k+1},Y_{k+2}, \dots, Y_n)$ for $i < n$, etc. Since the source is memoryless, X_k is independent of $\underline{X}_k^-$, so $I(X_k;J,\underline{Y}|\underline{X}_k^-) = I(X_k;J,\underline{Y},\underline{X}_k^-) \ge I(X_k;J,\underline{Y})$. Also Y_k -o- X_k -o- $(\underline{X}_k^-,\underline{X}_k^+,J,\underline{Y}_k^-)$, so $I((\underline{X};Y_k|J,\underline{Y}_k^-) = I(X_k;Y_k|J,\underline{Y}_k^-)$. Thus,

$$R \ge n^{-1} \sum_{k=1}^{n} I(X_k;J,\underline{Y}) - I(X_k;Y_k|J,\underline{Y}_k^-)$$

$$= n^{-1} \sum_{k=1}^{n} I(X_k;J,\underline{Y}_k^-) + I(X_k;\underline{Y}_k^+|J,\underline{Y}_k^-,Y_k)$$

$$= n^{-1} \sum_{k=1}^{n} I(X_k;W_k) + I(X_k;U_k|W_k,Y_k)$$

where $W_k \triangleq (J,\underline{Y}_k^{-1})$ and $U_k \triangleq \underline{Y}_k^+$. Note that Y_k -o- X_k -o- (W_k,U_k). It follows via standard arguments invoking the convexity of $R_0(\underline{D})$ that

$R \geq R_0(\underline{D})$, and therefore that $R(\underline{D}) \geq R_0(\underline{D})$. This completes the proof of the converse.

4. GENERALIZATION TO SEVERAL INFORMED DECODERS

Let $(\mathcal{X}, \mathcal{Y}_1, \mathcal{Y}_2, \cdots, \mathcal{Y}_m, p(x, y_1, y_2, \cdots, y_m))$ be a discrete memoryless multisource with generic random variables $X, Y_1, Y_2, \cdots, Y_m$. For $1 \leq i \leq m$ let $\mathcal{X}_i$ be a reconstruction alphabet and let

$$d_i : \mathcal{X} \times \hat{\mathcal{X}}_i \rightarrow [0, \infty)$$

be a distortion measure. An $(n, M, \underline{D})$ code consists of an encoder

$$f : \mathcal{X}^n \rightarrow \{0, 1, \cdots, M-1\}$$

and decoders $\underline{g} = (g_1, g_2, \cdots, g_m)$ satisfying

$$g_i : \{0, 1, \cdots, M-1\} \times \mathcal{Y}_i^n \rightarrow \hat{\mathcal{X}}_i \quad (1 \leq i \leq m)$$

The expected distortion $\underline{D} = (D_1, \cdots, D_m)$ for the code $(f, \underline{g})$ is given by

$$D_i = Ed_i(\underline{X}, \hat{\underline{X}}_i) \triangleq \frac{1}{n} \sum_{k=1}^{n} d_i(X_k, \hat{X}_{ik})$$

where

$$\hat{\underline{X}}_i = g_i(f(\underline{X}), \underline{Y}_i)$$

The rate R is $\underline{D}$-admissible if for every $\epsilon > 0$ there exists for some n an $(n, M, \underline{D}+\epsilon)$ code with $n^{-1} \log M \leq R + \epsilon$. The rate-distortion function is defined by

$$R(\underline{D}) = \inf\{R : R \text{ is } \underline{D}\text{-admissible}\}.$$

Let $P(\underline{D})$ denote the set of all discrete random variables $(U_T, \emptyset \neq T \subset \{1, \cdots, m\})$ jointly distributed with the generic source variables $(X, \underline{Y})$ such that the following two conditions are satisfied, where $V_T = (U_S : S \supset T, S \neq T)$:

(i) $\underline{Y} -\!\circ\!- X -\!\circ\!- (U_T, \emptyset \neq T \subset \{1, \cdots, m\})$

(ii) there exists $\hat{X}_i(V_{\{i\}}, U_{\{i\}}, Y_i)$ such that $Ed_i(X, \hat{X}_i) \leq D_i, 1 \leq i \leq m$.

Theorem 2. $R(\underline{D}) \leq R_0(\underline{D})$, where

$$R_0(\underline{D}) \triangleq \min_{P(\underline{D})} \sum_{\emptyset \neq T \subset \{1, \cdots, m\}} \max_{i \in T} I(X; U_T | Y_i, V_T)$$

Proof. One may employ a multiterminal random coding argument which, albeit involved, nonetheless can be considered relatively straightforward by today's standards [3]. In the interest of conservation and tedium-avoidance, we omit the details.

We have not been able to prove the converse of Theorem 2 for the general case. However, in the case in which the generic variables of the multisource satisfy the degradedness condition

$$X -\!\circ\!- Y_1 -\!\circ\!- Y_2 -\!\circ\!- \cdots -\!\circ\!- Y_m$$

we can show that $R(\underline{D}) = R_0(\underline{D})$. First, let us note the simplifications that result from said degradedness. If $i < j$ then decoder i can block-recover any $\underline{U}_T$ that decoder j can recover. This implies that there are only m non-trivial auxiliary random variables, namely $U_{\{1\}}, U_{\{1,2\}}, \ldots$ and $U_{\{1,\ldots,m\}}$. For simplicity of notation we denote these subsequently by $W_1, W_2, \ldots,$ and W_m, respectively. It is easy to see that for $1 \le i \le m$

$$V_{\{1,\ldots,i\}} = (W_{i+1}, \ldots, W_m) \text{ and } (V_{\{i\}}, U_{\{i\}}) = (W_i, W_{i+1}, \ldots, W_m).$$

It follows that in the degraded case

$$R_0(\underline{D}) = \min_{P(\underline{D})} \sum_{i=1}^{m} I(X; W_i | Y_i, W_{i+1}, \ldots, W_m)$$

where $P(\underline{D})$ is the set of all $\underline{W}$ jointly distributed with $(X, \underline{Y})$ such that

(i) $\underline{Y}$ -o- X -o- $\underline{W}$

(ii) there exists $\hat{X}_i(W_i, W_{i+1}, \ldots, W_m, Y_i)$ such that $Ed_i(X, \hat{X}_i) \le D_i$, $1 \le i \le m$.

In deducing this simplified form of $R_0(\underline{D})$, we have used the fact that the degradedness condition and condition (i) together imply that the maximum of $I(X; W_i | Y_i, W_{i+1}, \ldots, W_m)$ over $j \le i$ occurs for $j = i$.

Remark: In order to imbed the problem treated in Sections 2, and 3 into the more general problem currently under consideration, make the following associations: $m = 2$, $Y_1 = Y$, $Y_2 = \text{const.}$, $W_1 = U$, and $W_2 = W$.

Theorem 3. For the degraded multisource with X -o- Y_1 -o- Y_2 -o- $\ldots$ -o- Y_m, we have $R(\underline{D}) = R_0(\underline{D})$.

Proof. From Theorem 2 we know that $R(\underline{D}) \le R_0(\underline{D})$. Given an $(n, M, \underline{D})$ code, let $J = f(\underline{X})$ denote the output of the encoder. Let $\underline{Y}_i = (Y_{i,j}, 1 \le j \le n)$, $\underline{Y}^-_{i,k} = (Y_{i,j}, 1 \le j < k)$ and $\underline{Y}^+_{i,k} = (Y_{i,j}, k < j \le n)$; whenever no ambiguity results, contract notation from $\underline{Y}^{\pm}_{i,k}$ to $\underline{Y}^{\pm}_i$. We shall show that $n^{-1} \log M \ge R_0(\underline{D})$.

$$nR \triangleq \log M \ge H(J) \ge I(\underline{X}; J | \underline{Y}_m)$$

$$= I(\underline{X}; J, \underline{Y}_1, \underline{Y}_2, \ldots, \underline{Y}_{m-1} | \underline{Y}_m) - I(\underline{X}; \underline{Y}_{m-1} | J, \underline{Y}_m)$$

$$- I(\underline{X}; \underline{Y}_{m-2} | J, \underline{Y}_{m-1}, \underline{Y}_m) - \ldots - I(\underline{X}; \underline{Y}_1 | J, \underline{Y}_2, \ldots, \underline{Y}_m)$$

$$= \sum_{k=1}^{n} I(X_k; J, \underline{Y}_1, \underline{Y}_2, \ldots, \underline{Y}_{m-1} | \underline{Y}_m, \underline{X}^-) - I(X; Y_{m-1,k} | J, \underline{Y}^-_{m-1}, \underline{Y}_m)$$

$$- I(\underline{X}; Y_{m-2,k} | J, \underline{Y}^-_{m-2}, \underline{Y}_{m-1}, \underline{Y}_m) - \ldots - I(\underline{X}; Y_{1,k} | J, \underline{Y}^-_1, \underline{Y}_2, \ldots, \underline{Y}_m)$$

Since $(X_k, Y_{m,k})$ is independent of $(\underline{X}^-_k, \underline{Y}^-_{m,k}, \underline{Y}^+_{m,k})$, we have

$$I(X_k; J, \underline{Y}_1, \underline{Y}_2, \ldots, \underline{Y}_{m-1} | \underline{Y}_m, \underline{X}^-)$$

$$= I(X_k; J, \underline{Y}_1, \underline{Y}_2, \ldots, \underline{Y}_{m-1}, \underline{Y}^-_m, \underline{Y}^+_m, \underline{X}^- | Y_{m,k})$$

$$\ge I(X_k; J, \underline{Y}_1, \underline{Y}_2, \ldots, \underline{Y}_{m-1}, \underline{Y}^-_m, \underline{Y}^+_m | Y_{m,k})$$

The Markov string

$$Y_{m-1,k} \text{ -o- } (X_k, Y_{m,k}) \text{ -o- } (\underline{X}^-, \underline{X}^+, J, \underline{Y}^-_{m-1}, \underline{Y}^-_m, \underline{Y}^+_m)$$

implies

$$I(\underline{X}; Y_{m-1,k} | J, \underline{Y}^-_{m-1}, \underline{Y}_m) = I(X_k; Y_{m-1,k} | J, \underline{Y}^-_{m-1}, \underline{Y}_m).$$

More generally, for $1 \le i \le m-1$ the Markov string

$$\begin{aligned} Y_{i,k} &\text{ -o- } (X_k, Y_{i+1,k}, Y_{i+2,k}, \ . \ . \ . \ , Y_{m,k}) \\ &\text{ -o- } (\underline{X}^-, \underline{X}^+, J, \underline{Y}^-_i, \underline{Y}^-_{i+1}, \underline{Y}^+_{i+1}, \ . \ . \ . \ , \underline{Y}^-_m, \underline{Y}^+_m) \end{aligned}$$

implies

$$I(\underline{X}; Y_{i,k} | J, \underline{Y}^-_i, \underline{Y}_{i+1}, \underline{Y}_{i+2}, \ . \ . \ . \ , \underline{Y}_m) = I(X_k; Y_{i,k} | J, \underline{Y}^-_i, \underline{Y}_{i+1}, \underline{Y}_{i+2}, \ . \ . \ . \ , \underline{Y}_m)$$

Thus

$$\begin{aligned} R \ge \frac{1}{n} \sum_{k=1}^{n} \ & I(X_k; J, \underline{Y}_1, \underline{Y}_2, \ . \ . \ . \ , \underline{Y}_{m-1}, \underline{Y}^-_m, \underline{Y}^+_m | Y_{m,k}) \\ & - I(X_k; Y_{m-1,k} | J, \underline{Y}^-_{m-1}, \underline{Y}_m) \\ & - I(X_k; Y_{m-2,k} | J, \underline{Y}^-_{m-2}, \underline{Y}_{m-1}, \underline{Y}_m) \\ & \quad \vdots \\ & - I(X_k; Y_{1,k} | J, \underline{Y}^-_1, \underline{Y}_2, \ . \ . \ . \ , \underline{Y}_m) \\ = \frac{1}{n} \sum_{k=1}^{n} \ & I(X_k; J, \underline{Y}^-_{m-1}, \underline{Y}^-_m, \underline{Y}^+_m | Y_{m,k}) \\ & + I(X_k; \underline{Y}_1, \underline{Y}_2, \ . \ . \ . \ , \underline{Y}_{m-2}, \underline{Y}^+_{m-1} | Y_{m-1,k}, J, \underline{Y}^-_{m-1}, \underline{Y}_m) \\ & - I(X_k; Y_{m-2,k} | J, \underline{Y}^-_{m-2}, \underline{Y}_{m-1}, \underline{Y}_m) \\ & \quad \vdots \\ & - I(X_k; Y_{1,k} | J, \underline{Y}^-_1, \underline{Y}_2, \ . \ . \ . \ , \underline{Y}_m) \end{aligned}$$

Again, the first negative term in the summation may be canceled by a portion of the positive term immediately preceding it. Proceeding in this manner yields

$$R \geq \frac{1}{n} \sum_k I(X_k;J,\underline{Y}^-_{m-1},\underline{Y}^-_m,\underline{Y}^+_m|Y_{m,k})$$

$$+ I(X_k;\underline{Y}^-_{m-2},\underline{Y}^+_{m-1}|Y_{m-1,k},J,\underline{Y}^-_{m-1},\underline{Y}_m)$$

$$+ I(X_k;\underline{Y}^-_{m-3},\underline{Y}^+_{m-2}|Y_{m-2,k},J,\underline{Y}^-_{m-2},\underline{Y}_{m-1},\underline{Y}_m)$$

$$\vdots$$

$$+ I(X_k;\underline{Y}^+_1|Y_{1,k},J,\underline{Y}^-_1,\underline{Y}_2,\ \cdots\ ,\underline{Y}_m)$$

Degradedness gives the Markov string

$$Y_{m,k} \text{ -o- } Y_{m-1,k} \text{ -o- } (X_k,J,\underline{Y}^-_{m-1},\underline{Y}^-_m,\underline{Y}^+_m),$$

which implies that

$$I(X_k;\underline{Y}^-_{m-2},\underline{Y}^+_{m-1}|Y_{m-1,k},J,\underline{Y}^-_{m-1},\underline{Y}_m)$$

$$= I(X_k;\underline{Y}^-_{m-2},\underline{Y}^+_{m-1}Y_{m,k}|Y_{m-1,k}J,\underline{Y}^-_{m-1},\underline{Y}^-_m,\underline{Y}^+_m)$$

$$\geq I(X_k;\underline{Y}^-_{m-2},\underline{Y}^+_{m-1}|Y_{m-1,k},J,\underline{Y}^-_{m-1},\underline{Y}^-_m,\underline{Y}^+_m)$$

Similarly, degradedness yields

$$(Y_{m-1,k},Y_{m,k}) \text{ -o- } Y_{m-2,k} \text{ -o- } (X_k,J,\underline{Y}^-_{m-2},\underline{Y}^-_{m-1},\underline{Y}^+_{m-1},\underline{Y}^-_m,\underline{Y}^+_m)$$

which implies that

$$I(X_k;\underline{Y}^-_{m-3},\underline{Y}^+_{m-2}|Y_{m-2,k},\underline{Y}^-_{m-2},\underline{Y}_{m-1},\underline{Y}_m)$$

$$\geq I(X_k;\underline{Y}^-_{m-3},\underline{Y}^+_{m-2}|Y_{m-2,k},J,\underline{Y}^-_{m-2},\underline{Y}^-_{m-1},\underline{Y}^+_{m-1},\underline{Y}^-_m,\underline{Y}^+_m)$$

We continue iteratively in this vein until finally observing that the Markov string

$$Y_{2,k},Y_{3,k},\ \cdots\ ,Y_{m,k} \text{ -o- } Y_{1,k} \text{ -o- } X_k,J,\underline{Y}^-_1,\underline{Y}^-_2,\underline{Y}^+_2,\ \cdots\ ,\underline{Y}^-_m,\underline{Y}^+_m$$

implies

$$I(X_k;\underline{Y}^+_1|Y_{1,k},J,\underline{Y}^-_1,\underline{Y}^-_2,\underline{Y}^+_2,\ \cdots\ ,\underline{Y}^-_m,\underline{Y}^+_m)$$

Thus, if we set

$$W_{m,k} = J,\underline{Y}^-_{m-1},\underline{Y}^-_m,\underline{Y}^+_m$$

$$W_{m-1,k} = \underline{Y}^-_{m-2},\underline{Y}^+_{m-1}$$

$$W_{m-2,k} = \underline{Y}^-_{m-3},\underline{Y}^+_{m-2}$$

$$\vdots$$

$$W_{1,k} = \underline{Y}^+_1,$$

we may write

$$R \geq \frac{1}{n}\sum_{k=1}^{n} I(X_k;W_{m,k}|Y_{m,k}) + I(X_k;W_{m-1,k}|Y_{m-1,i},W_{m,k})$$

$$+ I(X_k;W_{m-2,k}|Y_{m-2,k},W_{m-1,i},W_{m,i})$$

$$+ \cdots + I(X_k;W_{1,k}|Y_{1,k},W_{2,k},\ldots,W_{m,k})$$

$$= \frac{1}{n}\sum_{k=1}^{n}\sum_{i=1}^{m} I(X_k;W_{i,k}|Y_{i,k},W_{i+1,k},\ldots,W_{m,k})$$

Observe that

$$(Y_{i,k}, 1 \leq i \leq m) \text{ -o- } X_k \text{ -o- } (W_{i,k}, 1 \leq i \leq m) = (J,\underline{Y}^+_1,\underline{Y}^-_2,\underline{Y}^+_2,\ldots,\underline{Y}^-_m,\underline{Y}^+_m)$$

so the $W_{i,k}$ satisfy (i) in the definition of $P(\underline{D})$ for each $1 \leq k \leq n$. Also, $(W_{i,k}, W_{i+1,k}, \ldots, W_{m,k}, Y_{i,k}) = (\underline{Y}^-_{i-1},\underline{Y}^+_i,\underline{Y}^-_i,\underline{Y}^+_{i+1},\ldots,\underline{Y}^-_{m-2},\underline{Y}^+_{m-1},J,\underline{Y}^-_{m-1},\underline{Y}^-_m,\underline{Y}^+_m,Y_{i,k})$ contains $(J,\underline{Y}_i)$, which is the totality of information available to decoder i. Hence, there exists $\hat{X}_{i,k}(W_{i,k},\ldots,W_{m,k}, Y_{i,k})$ such that $E\, d_i(X_k, \hat{X}_{i,k}) \leq D_{i,k}$, where $D_{i,k}$ is the average distortion with which decoder i reproduces X_k. It follows that

$$R \geq \frac{1}{n}\sum_{k=1}^{n} R_0(\underline{D}_k)$$

where $\underline{D}_k = (D_{1,k},\ldots,D_{m,k})$. An argument similar to that employed in Section 3 shows that $R_0(\cdot)$ is convex, so

$$R \geq R_0\left(\frac{1}{n}\sum_{k=1}^{n}\underline{D}_k\right) = R_0(\underline{D}).$$

Theorem 3 is proved.

The degradedness condition X -o- Y_1 -o- Y_2 -o- $\cdots$ -o- Y_m need not be physical. Also, $R(\underline{D})$ clearly depends on $p(x,\underline{y})$ only through its second-order marginals $\{p(x,y_i), 1 \leq i \leq m\}$.

5. THE QUADRATIC GAUSSIAN CASE

Suppose $Y = X + Z$, where X and Z are independent, (zero mean) Gaussian r.v. with respective variances σ_X^2 and σ_Z^2. Since the source alphabets X and Y no longer are finite, a formal justification of the extension of the results to the case at hand is in order. We omit this justification, since it directly parallel's Wyner's [6] extension of his work with Ziv [1]. The minimization defining $R(D_1,D_2)$ can be carried out explicitly in this case. Not surprisingly, it occurs when $U = X + Z_1$ and $W = X + Z_2$ with Z, Z_1, and Z_2 all independent, zero mean and Gaussian; i.e., U, W and Y correspond to observations of X via independent channels with additive Gaussian noise. Decoder 1 forms the linear combination of all three of them,

$$\hat{X}_1 = \alpha U + \beta W + \gamma Y,$$

that is its optimum estimate of X given (U, W, Y). Decoder 2 sees only W and therefore forms

$$\hat{X}_2 = \delta W$$

It is easy to see that the minimum of $I(X;W) + I(X;U|YW)$ subject to $E(X-\hat{X}_i)^2 = D_i (i=1,2)$ occurs when

$$I(X;W) = I(X;\hat{X}) = \frac{1}{2}\ln(\sigma_X^2/D_2)$$

and

$$I(X;U|YW) = I(X;\hat{X}_1|YW) = \frac{1}{2}\ln\left(\frac{\sigma^2(X|YW)}{\sigma_1}\right)$$

where $\sigma^2(X|YW)$ is the conditional variance of X given Y and W. Knowledge of Y as well as W reduces the conditional variance of X from its value D_2, when only W is known, to $D_2\sigma_Z^2/(D_2+\sigma_Z^2)$. It follows that

$$R(D_1,D_2) = \begin{cases} \frac{1}{2}\ln\left(\frac{\sigma_x^2\sigma_Z^2}{D_1(\sigma_x^2+\sigma_Z^2)}\right) \text{if } D_1 \le \frac{D_2\sigma_Z^2}{D_2+\sigma_Z^2} \\ \frac{1}{2}\ln(\sigma_x^2/D_2) \quad \text{if } D_1 \ge \frac{D_2\sigma_Z^2}{D_2+\sigma_Z^2} \end{cases}$$

6. BINARY HAMMING EXAMPLE

Now suppose that X is the input to and Y the output of a hypothetical binary symmetric channel (BSC) of crossover probability $p < 1/2$. Assume that $P(X = 0) = P(X = 1) = 1/2$ and that distortion is measured by Hamming distance, $d(x,\hat{x}) = (x+\hat{x}) \bmod 2$. Let the r.v. U' be the result of passing X through another BSC with crossover probability β that operates independently of that which connects X and Y, where $\beta \le D_0 \triangleq \min(p,D_2)$. Let U be the result of passing U' through a binary erasure channel (BEC) with erasure probability e that operates independently of both the aforementioned BSC's. Finally, let W be the result of passing U' through another independent BSC whose crossover probability is such that $Ed(X,W) = D_2$ (See Figure 2).

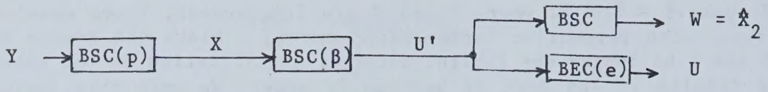


Figure 2. Block diagram depicting joint distribution of (X,Y,U,W)

It is clear that the requirement Y -o- X -o- (U,W) is satisfied. Moreover, the $\hat{X}_1(U,W,Y)$ that minimizes $E\,d(X,\hat{X}_1)$ is

$$\hat{X}_1 = \begin{cases} U & \text{if no erasure occurs} \\ W & \text{if an erasure occurs and } D_2 < p \\ Y & \text{if an erasure occurs and } D_2 \geq p, \end{cases}$$

and the minimized average distortion that results is

$$D_1 = (1-e)\beta + e \min(p,D_2).$$

For given (D_1,D_2) and β this equation specifies $e = e(\beta)$. When we evaluate $I(X;W) + I(X;U|WY)$ it equals

$$1 - h(D_2) + [1-e(\beta)][g(\beta)-g(D_2)]$$

where $h(x)$ is the binary entropy function, $g(x) = h(p*x) - h(x)$, and $p*x = p(1-x) + (1-p)x$. It follows that

$$R(D_1,D_2) \leq 1-h(D_2) + \min_{\beta}[1-e(\beta)][g(\beta)]-g(D_2)].$$

The minimum is over $\beta \in [0,\min(p,D_2)]$, and $e(\beta)$ is specified by the above equation for D_1.

We believe that $R(D_1,D_2)$ actually equals the right hand side of the previous equation, but we have not yet proved that the choice of (W,U) in question is the optimum one in $P(\underline{D})$. In the special cases that correspond to ordinary rate-distortion and to the Wyner-Ziv problem, our candidate expression reduces to the known correct answer.

7. REFERENCES

[1] A. D. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Trans. Inform. Theory*, vol. IT-22, pp. 1-10, January 1976.

[2] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 471-480, July 1973.

[3] I. Csiszar and J. Korner, *Information Theory: Coding Theorems for Discrete Memoryless Systems*. Academic Press: New York, and Akademia Kiado: Budapest, 1981.

[4] A. D. Wyner, "On source coding with side information at the decoder," *IEEE Trans. Inform. Theory*, vol. IT-21, pp. 163-179, 1975.

[5] T. Berger, "Multiterminal source coding." In the *Information Theory Approach to Communications* (G. Longo, ed.), CISM Courses and Lecture Notes No. 229, Springer, Wien-New York, pp. 172-231, 1977.

[6] A. D. Wyner, "The Rate-Distortion Function for Source Coding with Side Information at the Decoder-II: General Sources," *Information and Control* vol. 38, pp. 60-80, 1978.

THE CONTROL OF ROBOT MANIPULATORS USING AN OPTIMAL DECISION STRATEGY

MARK W. SPONG SHAYGAN KHERADPIR JAMES S. THORP

School of Electrical Engineering, Cornell University,
Ithaca, NY 14853

ABSTRACT

The problem of following a desired trajectory in the state space of an n-link robotic manipulator with bounds on the allowable input torque is considered. Using a so-called Optimal Decision Strategy (ODS), a pointwise optimal control law is derived which at each time t minimizes the deviation between the system velocity vector and a desired velocity vector, subject to the input constraints. The design of the optimal control law is reduced to the solution of a quadratic programming problem which is solved using the Primal-Dual Method. It is shown that by various choices of the desired velocity vector, our control law specializes to several well-known control laws when the restriction on the input torque is removed. These include the computed torque method of Bejczy[2], and Paul[9], as well as the control laws derived by Saridis and Lee [10], Freund [4], and others [5],[11]. Simulations are presented for the case of a two-link manipulator with bounded input and our results are compared with the computed torque method.

I. INTRODUCTION

Computer controlled robotic manipulators are beginning to have a major impact on the nature of manufacturing systems. By replacing workers in dull, repetitive, or hazardous environments while simultaneously increasing productivity, robots are finding ever expanding applications in many areas of manufacturing. This trend toward automation through the introduction of robots will undoubtedly accelerate in the future.

Most robots currently manufactured control each joint of the arm independently using a constant gain linear servo-system (e.g. a PID regulator) and treat coupling effects, uncertainties, friction, other nonlinearities, etc. as disturbances which are to be rejected by the controller. The simplicity of such controllers enhances the robustness of the system in the face of unmodeled dynamics but in doing so some sacrifice in performance inevitably results. For applications involving high speeds such simple controllers require large amounts of power which waste energy and which may exceed the capabilities of the actuators resulting in unacceptable position and velocity errors with unnecessary vibration. The result is that present day robots are primarily suited for tasks such as materials transfer or spot welding which do not require large fast motions but which require merely steady state position accuracy and a high degree of repeatability. Since precise, high speed operation will be required of advanced robots of the future it is clear that more sophisticated control algorithms will also be required.

There have been several schemes proposed for the design of nonlinear feedback controllers for n-link robot manipulators [2-4,8-10]. A common drawback shared by virtually all of these methods is that they too may require large magnitudes for the input torque which, in fact, is not taken

---

This research was supported by a grant from the Cornell Manufacturing Engineering Program (COMEP)

into account in the derivation of the control laws. Due to the high degree of coupling among joints and severe nonlinearities in the system behavior input constraints must be considered, especially if one desires large or fast motions of the manipulator. Primarily for this reason we consider here the problem of tracking a desired trajectory in the state space of an n-link manipulator with bounds on the allowable input torques. Our controller is designed using a so-called Optimal Decision Strategy(ODS)[12],[13]. Based on the state space description of an n-link manipulator derived from the Lagrangian formulation of the equations of motion we are able to formulate the problem of trajectory following as a quadratic programming problem, where the Decision Strategy (objective) is to minimize the Euclidean norm of the difference between the actual instantaneous velocity of the system and a desired velocity vector which may be generated by a number of techniques that we discuss. The constraints in the quadratic programming problem arise due to the bounds on the allowable input torque. Solving the quadratic programming problem using standard techniques yields an on-line optimal control law for trajectory following in the face of input constraints. By various choices of the desired velocity vector we show that several well known algorithms in the robotics literature are special cases of our optimal control law if we allow unbounded control. These include the computed torque method, as well as the methods of Freund [4], Saridis and Lee [10], and others [5],[11]. Thus we view our results as providing a unifying framework for viewing a large class of nonlinear control schemes for robot manipulators. Finally we give some simulation results for a particular choice of the desired velocity vector and compare our method with the computed torque method.

II. BACKGROUND ON OPTIMAL DECISION STRATEGIES

Classical optimization techniques are "far sighted" in the sense that entire control-trajectory pairs are examined to obtain that pair which is optimal with respect to a desired criterion. For applications to robotics, where there may be a high collision risk unless tight constraints are imposed, one would prefer an instantaneous or pointwise optimization procedure which optimizes the present state of the system without regard to future events. Such "short sighted" optimization procedures, under the name Optimal Decision Strategies(ODS) [12], have been applied to the transient control problem for power systems. For discussion purposes suppose that the dynamical system under consideration is represented in state form as

$$\dot{x}(t) = f(x(t), u(t)) \tag{1}$$

$x(t) \in R^n$, $t \geq 0$, where the control input $u(t)$ must satisfy

$$u(t) \in \Omega \text{ , for } t \geq 0. \tag{2}$$

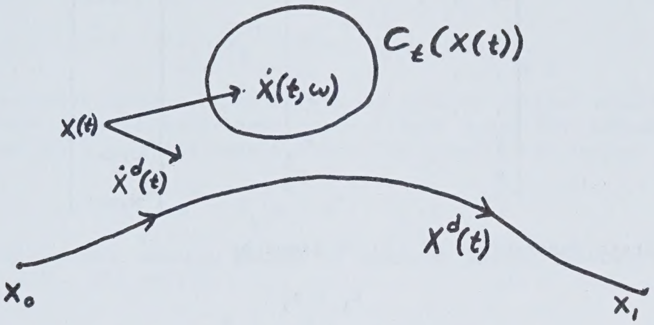
for some bounded subset $\Omega \subset R^m$.

Let $t \rightarrow x^d(t)$ define a trajectory in state space which we wish system (1) to track. Fix time $t \geq 0$ and consider the diagram of figure 1.

The constraint set C_t is defined to be the translate by $x(t)$ of the set

$$C(x(t)) = \{\dot{x} \in R^n \mid \dot{x} = f(x(t),\omega) \text{ , } \omega \in \Omega \}.$$

figure 1



Note that the set C is generated by the specific control constraint set Ω by way of equation (1).

We assume that at time t a desired velocity vector $\dot{x}^d$ is given. $\dot{x}^d$ is a function of both t and x(t) and is subject only to smoothness conditions necessary to ensure the existence of a solution to our optimization problem to follow. We shall show explicitly how to generate $\dot{x}^d(t)$ in the next section. The control law u(t) is then chosen so that at each t, $||\dot{x}(t) - \dot{x}^d(t)||$ is minimized subject to $\dot{x}(t) \in C_T$ (equivalently $u(t) \in \Omega$). It can be shown that the minimizing $\dot{x}(t)$ is unique whenever C_T (or Ω) is convex and f(x,u) is convex, one-to-one in u for every choice of $x \in \mathbb{R}^n$ [12].

III. MAIN RESULTS

Consider the equations of motion of an n-link manipulator derived from the Euler-Lagrange equations

$$D(q)\ddot{q} + H(q,\dot{q}) = u(t) \tag{3}$$

where $q(t) \in \mathbb{R}^n$ are the generalized coordinates representing joint positions, D(q) is an nxn matrix of acceleration related terms, $H(q, \dot{q})$ is an n-vector function representing centripetal, Coriolis, gravitational and frictional forces and torques, and $u(t) \in \mathbb{R}^n$ represents the input torque at each joint. We assume that the input torque is constrained via

$$|u_i(t)| \leq u_{i,\max} \qquad i=1,\ldots n \tag{4}$$

which we may write as

$$Gu \leq b \tag{5}$$

where the 2nxn matrix G and the 2nx1 column vector b are given by

$$G = \begin{bmatrix} 1 & 0 & . & . & . & 0 \\ -1 & 0 & . & . & . & 0 \\ . & . & & & & . \\ . & . & & & & . \\ . & . & & & & . \\ 0 & & & & & 1 \\ 0 & . & . & . & . & -1 \end{bmatrix}, \quad b = \begin{bmatrix} u_{1,max} \\ u_{1,max} \\ . \\ . \\ . \\ u_{n,max} \\ u_{m,max} \end{bmatrix} \tag{6}$$

The state description of (1) is given by

$$\dot{x}_1 = x_2 \tag{7}$$

$$\dot{x}_2 = f(x) + g(x)u \tag{8}$$

where $x(t) = (x_1(t), x_2(t)) = (q(t), \dot{q}(t))$ is the state at time t, and

$$f(x) = -D^{-1}(x_1)H(x_1, x_2) \qquad g(x) = D^{-1}(x_1) \tag{9}$$

For the moment let us assume that at each time t a desired velocity vector $\dot{x}^d(t)$ is specified (We shall discuss how to choose $\dot{x}^d$ later. We may assume that $\dot{x}^d(t)$ is at least piecewise continuous for $t > 0$.) Because of the form of (7-9) we assume that $\dot{x}^d$ has the form $\dot{x}^d = (x_2, \dot{x}_2^d)$ We then seek a control law such that along the actual trajectory of the system u(t) minimizes the Euclidean norm

$$||\dot{x}^d - \dot{x}||$$

subject to

$$Gu \leq b$$

By exploiting the special structure of (7-9) and minimizing the square of the above norm (which is an equivalent problem) the design of the optimal control law reduces to the quadratic programming problem

$$\min(\dot{x}_2^d - f(x) - g(x)u)^T(\dot{x}_2^d - f(x) - g(x)u) \tag{10}$$

subject to

$$Gu \leq b \tag{11}$$

This problem may be solved using standard techniques, for example the Primal-Dual method [7]. The optimal control takes the form

$$u(t) = (g^T g)^{-1}(g^T(\dot{x}_2{}^d - f) - G^T\lambda) \tag{12}$$

where the vector of Lagrange multipliers λ may be obtained from Hildreth's Quadratic Programming Procedure[7]. It should be remarked here that the form of our control law (12) is very similar to that derived in [12] in the context of large-scale power systems. Simplifying the control law (12) using equations (9) yields

$$u(t) = D(x_1)(\dot{x}_2^d - D(x_1)^T G^T \lambda) + H(x_1, x_2) \tag{13}$$

It is very interesting to note at this point that by various choice of $\dot{x}_2^d$ we may recover several well-known control laws under the assumption of unbounded control (in which case $\lambda = 0$). For example the choice

$$\dot{x}_2^d = \ddot{q}^d - K_v(\dot{q} - \dot{q}^d) - K_p(q - q^d) \tag{14}$$

yields the well-known inverse method or computed torque method of Bejczy [2] and Paul [9]. The choice

$$\dot{x}_2^d = S(x^d(t) - x(t)) \tag{15}$$

yields the controller developed by Saridis and Lee [10] where S was determined by a sequential design procedure. The choice

$$\dot{x}_2^d(t) = -\left[\Lambda_0, \Lambda_1\right]x(t) + \left[\Lambda_0, 0\right]x^d(t) \tag{16}$$

where

$$\Lambda_0 = \mathrm{diag}\left[\omega_1^2, \ . \ . \ . \ , \omega_n^2\right] \tag{17}$$

$$\Lambda_1 = \mathrm{diag}\left[2\zeta_1\omega_1, \ . \ . \ . \ , 2\zeta_n\omega_n\right] \tag{18}$$

yields the control scheme of Freund [4]. Thus we have shown that several of the best known control schemes are really special cases of a pointwise optimal control laws using the above optimal decision strategy. Thus is important in that it provides a common theoretical framework for these methods. Indeed, the well-known inverse problem of optimal control [6] is, given a control law, to find the criteria under which it becomes an optimal control law. In addition, using our results several other control laws, for example those of Hemani and Camana [5], and the control scheme of Takegaki and Arimoto [11], are seen to be pointwise optimal in the above sense. Perhaps more important, however, is the fact that none of the aforementioned methods treat the problem of constraints on the allowable input torque. Our treatment, on the other hand, handles constraints quite easily. In fact, constraints on the state (due to obstacles in the workplace, etc.) are also easily handled in this context.

IV. SIMULATION RESULTS

A two link manipulator was simulated using the SIMNON package from LUNDPAK [1]. The computed torque method was used to drive the manipulator so that both angles followed a fast, critically damped step response. The angles and torques are shown in Figures 2a and 2b. The rise time is fast enough (roughly 0.5 sec) that unreasonably large torques are required. If the computed torque method is saturated at a torque of approximately one seventh of the value from Figure 2b, the trajectories and torques of Figures 3a and 3b result. The resulting undershoot and overshoot in the angles are, of course, unacceptable for robotics applications.

Figures 4a and 4b show the response obtained using the pointwise strategy described in (10)-(13). $\dot{x}_2^d$ was generated from a family of critically damped step responses as follows:

At each sample instant t, $\dot{x}_2^d$ is chosen to be tangent to one member of the family of critically damped step responses shown in Figure 5. In other words

$$\dot{x}_2^d = a_1x_1(t) + a_2x_2(t) = a_1q(t) + a_2\dot{q}(t) \tag{19}$$

where a_1, a_2 are chosen so that the system

$$\ddot{y} + a_1\dot{y} + a_2y = u \tag{20}$$

has a critically damped step response, e.g., $a_1 = 20$, $a_2 = 100$.

Effectively, then, at each sample instant the current state $(q,\dot{q})$ becomes the initial condition for (20), such that, in the absence of further constraints or errors the system would follow that particular response from in Figure 5 on which the initial conditions place it. It can be seen from figures 4a and 4b that the response is somewhat slower because of the torque constraint but that (local) critical damping is maintained. It is clear that the pointwise strategy adjusts both torques in the interval when only one of the torques in Figure 1b violates the constraint in order to achieve the desired performance.

Computationally, it is known that the class of nonlinear control schemes to which our method belongs imposes a significant burden on the control computer for a 6-link manipulator[2],[8], and much recent effort has gone into the problem of deriving efficient methods for computing the terms $D(q)$ and $H(q,\dot{q})$ in (1) [8]. Using such techniques as the recursive Newton-Euler method [8], it now appears computationally feasible to implement these various schemes for a 6-link manipulator. For our method, the only additional computational burden is in the computation of the vector of Lagrange multipliers $\lambda(t)$. However, since the quadratic programming scheme that we use is rapidly convergent (3 or 4 iterations, in our simulations) and since most of the terms involved in the iterative scheme are by products of the computation of $D(q)$ and $H(q,\dot{q})$, the computation of $\lambda(t)$ does not impose significant additional burden on the control computer.

V. ACKNOWLEDGEMENT

The first author would like to aknowledge several helpful discussions with J. McClure and R.J. Thomas on the subjects of Robotics and Optimal Decision Strategies.

VI. REFERENCES

[1] Astrom, K.J., "Computer-Aided Modeling, Analysis and Design of Control Systems: A Perspective", IEEE Control Systems Magazine, Vol.3, No. 2, May 1983, pp. 4-16.

[2] Bejczy,A.C. "Robot Arm Dynamics and Control", JPL Technical Memorandum 33-669, Feb. 1974.

[3] Dubowsky, and DesForges, "The Application of Model Referenced

Adaptive Control to Robot Manipulators", J. Dyn. Sys. Meas. and Cont. ,101, (1979) pp. 193-200.

[4] Freund, "Fast Nonlinear Control with Arbitrary Pole-Placement for Industrial Robots and Manipulators", Int. J. Robotics Res., V.1 , No. 1, (1982) pp. 65-78.

[5] Hemami, H. and Camana, P.C., "Nonlinear Feedback in Simple Locomotion Systems", IEEE Trans. Aut. Cont., AC-21, No.6, Dec. 1976, pp. 855-860.

[6] Kalman, R.E., "When is a Control System Optimal", Trans. ASME, J. Basic Eng., Vol. 86, Mar. 1964, pp. 51-60.

[7] Luenberger, Optimization by Vector Space Methods, John Wiley and Sons, Inc., New York, 1969.

[8] Luh, Walker, and Paul, "On-line Computational Schemes for Mechanical Manipulators", J. Dyn. Sys. Meas. and Cont., 102 (1980) pp. 69-76.

[9] Paul, "Modeling, Trajectory Calculation, and Servoing of a Computer Controlled Arm", Stanford University, AIM 177, Nov. 1972.

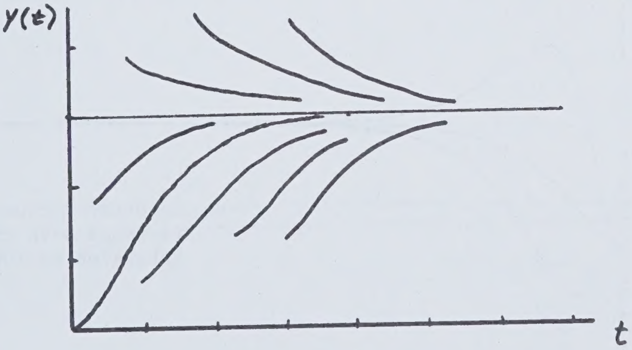
[10] Saridis,G.N., and Lee, C.S.G., "Approximation Theory of Optimal Control for Trainable Manipulators", IEEE Trans. Syst. Man, Cyber., SMC-8, Mar. 1979.

[11] Takegaki, and Arimoto, "A New Feedback Method for Dynamic Control of Manipulators", J. Dyn. Sys. Meas. and Cont., 102 (1981) pp. 119-125.

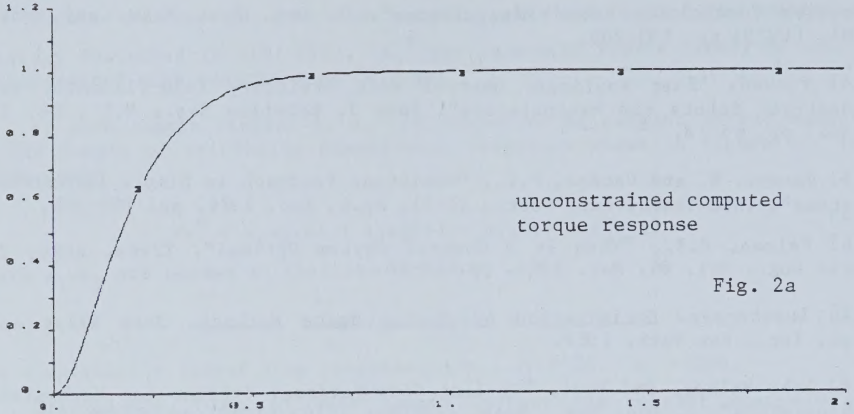
[12] Thomas,R.J., "On the Control of Transient Modes in Large Electric Power Systems", Proc. 8th Annual Southeastern Symposium on System Theory, 1976, pp 197-202.

[13] Thomas,R.J., Thorp, and Pottle, "A Model-Referenced Controller for Stabilizing Large Transient Swings in Power Systems", IEEE Trans. Aut. Cont. AC-21, no. 5 Oct. 1976, pp 746-750.

Figure 5

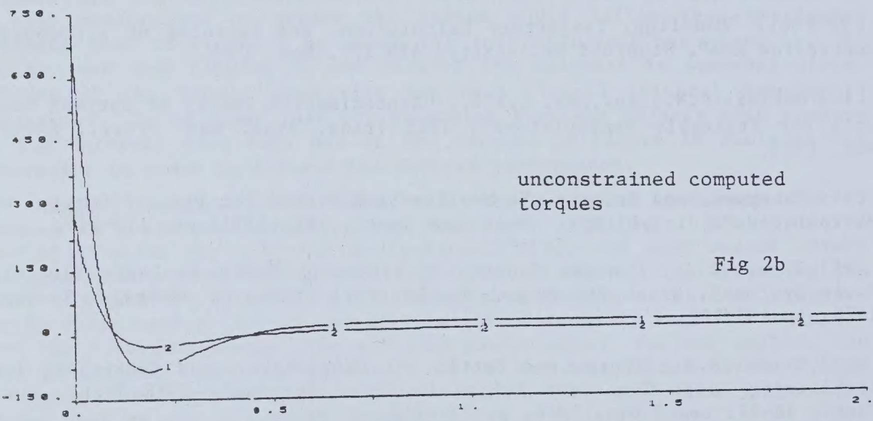


Family of critically damped step responses



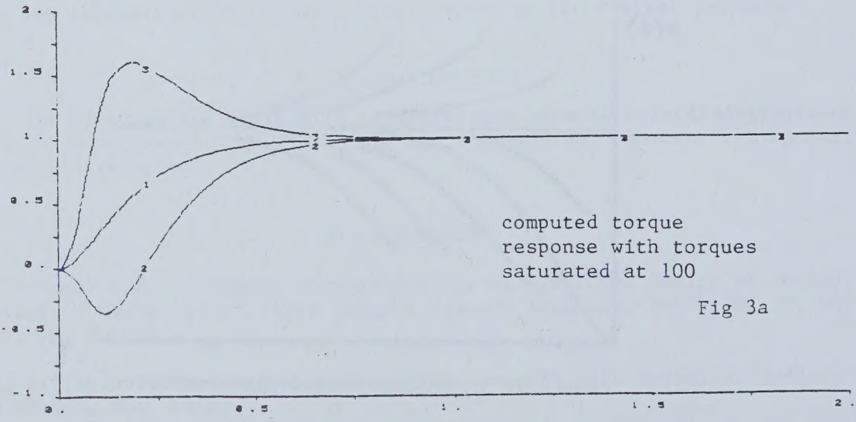
unconstrained computed torque response

Fig. 2a



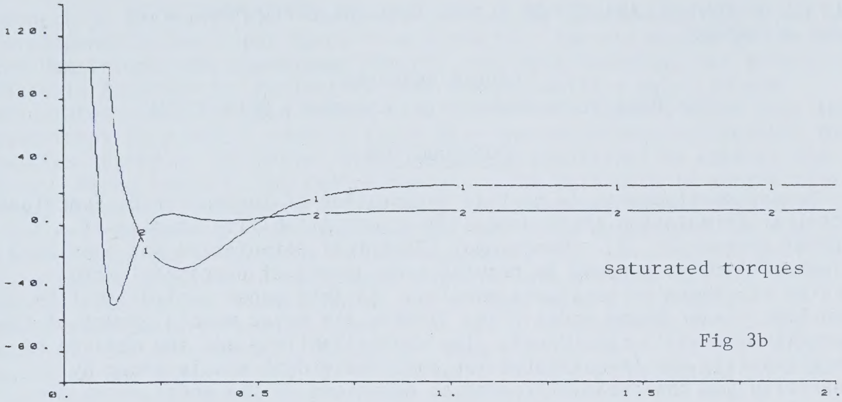
unconstrained computed torques

Fig 2b



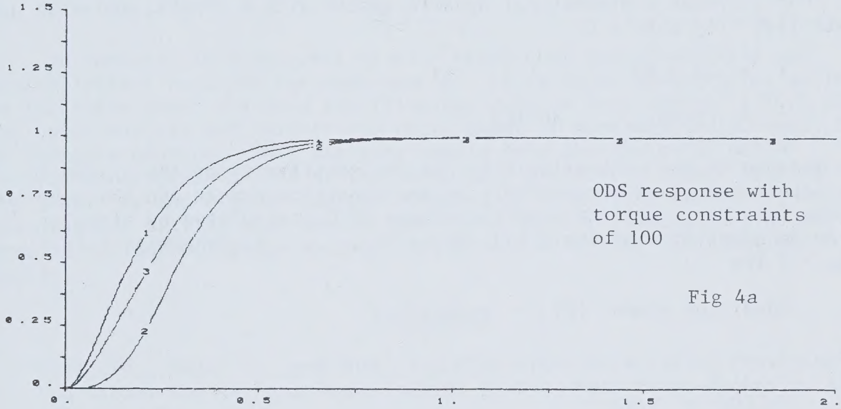
computed torque response with torques saturated at 100

Fig 3a



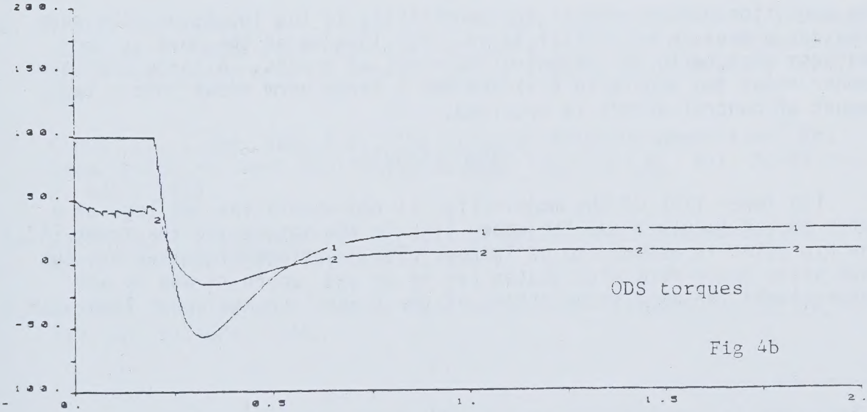
saturated torques

Fig 3b



ODS response with torque constraints of 100

Fig 4a



ODS torques

Fig 4b

ANALYSIS OF CONTROLLABILITY OF PLANAR TWO-LINK BIPED MODEL WITH MUSCLE ACTUATORS

BEHNAM BAVARIAN
The Ohio State University, Columbus, Ohio 43210

INTRODUCTION

Recent developments in partial restoration of locomotion by Functional Electrical Stimulation (FES) necessitate controllability studies of skeletal systems [1, 2]. Functional Electrical stimulation has been used for rehabilitative purpose to provide some degree of purposeful motor function in otherwise paralyzed muscles. In this paper controllability of a two-link planar biped model (Fig. 1) with six major muscle groups of the lower extremities is considered. The controllability and the measure of controllability are investigated for each individual muscle group by linearizing the nonlinear differential equations of the model about the vertical stance.

For a linear n-dimensional dynamic system with m inputs, and with controllability matrix C:

$$X = AX + BU$$

$$C = [B | AB | \ . \ . \ . \ A^{n-1}B]$$

The measure of the controllability and the sensitivity of the system to the inputs are given respectively by the condition number and the norm-two of the C matrix. Let the singular values of C, calculated by singular value decomposition of the C [3], be $\lambda_1, \lambda_2, \dots, \lambda_n$, where $\lambda_1 > \lambda_2 > \dots > \lambda_n > 0$ the

$$\text{condition number } (C) = \lambda_1/\lambda_n$$

and

$$||C||_2 = 1/\lambda_n$$

The condition number governs the sensitivity to the inputs and the norm provides a measure of control input. The inverse of the norm is the nearness measure to the uncontrollable set of states. A large condition number shows too sensitive a system and a large norm shows that a large amount of control effort is required.

DISCUSSION

The lower link of the model (Fig. 1) represents the two legs of a human locked together and the upper link is the thighs and the trunk [4]. The hip joint is assumed to be locked. The equations of motion are put into state space form with states $[\dot{\theta}_1 \ \dot{\theta}_2 \ \theta_1 \ \theta_2]$, where θ_1 and θ_2 are respectively the angular positions of the lower link and upper link with

---

* This work is supported in part by NSF grant ECS-8201240, and in part by the Dept. of Elec. Engr., The Ohio State University.

respect to the vertical. The B matrix is a 4x6 matrix each column of which corresponds to one input force from a muscle. The six muscle groups are the Hamstrings, the Quadriceps Femoris, the Gastrocnemius, the Soleus, the Tibialis Anterior and Posterior. Six controllability matrices are calculated - one for each column of B (Table 1). Table 1 shows that the Quadriceps is a better control input than the Hamstrings and further those muscles acting on the ankle joint are better positioned to control the body. Among the six, the Gastrocnemius is the best both in sensitivity and input measure. The Gastrocnemius is a two joint muscle and this result confirms the general consensus that the two joint muscle provide energy conservation and efficiency in body movements [5]. In order to prolong the duration of standing in paraplegic patients and not tire out the stimulated muscle alternative stimulation of two or more muscle is proposed [2]. Here three such postures are analyzed (Fig. 2). The suggested stimulated muscles, the corresponding condition number and norm of the controllability matrices are shown in Fig. 2. These multi-muscle stimulations show much less sensitivity and control effort compared to the single muscle stimulations of Table 2. Among the latter three the third one has the least sensitivity and comparable control effort requirement with respect to the other two.

In summary, this analysis gives a measure of the sensitivity and control effort required for each muscle. It is shown that muscles acting on the ankle joint are less sensitive and require less control effort than the upper muscles and further the recruitment of more muscle increases the performance measure. This analysis should help the design of better control strategies for FES of the paraplegic patient. Further studies in the stability and digital computer simulation of certain movements, such as standing and sitting down are in order here for verification of the present results and advancing the state of Functional Electrical Stimulation of muscle.

REFERENCES

1. Kralj, A., Bajd, T., and Turk, R., Electrical Stimulation Providing Functional Use of Paraplegic Patient Muscle, Med. Prog. Technol., 1980, vol. 7, 3-9.

2. Kralj, A., Jaeger, R.J., Bajd, T., Posture Switching Enables Prolonged Standing in Paraplegic Patients Functionally Electrically stimulated, 5th Annual Conference on Rehabilitation Engineering, Houston, Texas, 1982.

3. Klema, V.C., and Laub, A.J., The Singular Value Decomposition, Its Computation and Some Applications, IEEE Trans. on AC, Vol. AC-25, No. 2, April 1980.

4. Bavarian, B., Some Control Aspects of Multi-Link Rigid Bodies, Doc. Dis., The Ohio State University, 1984.

5. Elftman, H., The Work Done by Muscles in Running, Amer. J. Physiol. 129, pp. 672-684, 1940.

Table 1

C matrix	H	QF	S	TA	G	TP
Condition No.	30608	30608	3248	3248	1106	3248
Norm-2	8517	6254	818	1383	676	1569

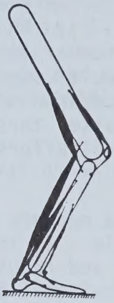
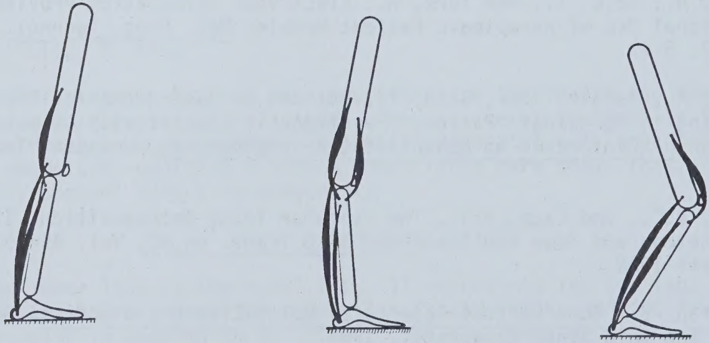


Fig. 1. The two-link model in sagittal plane. The foot is massless and H = Hamstrings QF = Quadriceps Femoris, G = Gastrocnemius S = Soleus, TA = Tibialis Anterior, TP = Tibialis Posterior.



$\theta_1 = -10°$, $\theta_2 = -10°$
Cond. No. = 78
Norm = 59

$\theta_1 = 5°$, $\theta_2 = -10°$
Cond. No. = 252
Norm = 56

$\theta_1 = -30°$, $\theta_2 = 20°$
Cond. No. = 29
Norm = 74

Fig. 2. Three different postures for prolonged standing.

A DECENTRALIZED VARIABLE STRUCTURE CONTROL ALGORITHM FOR ROBOTIC MANIPULATORS

RUSSELL G. MORGAN, ÜMIT ÖZGÜNER
The Ohio State University, Columbus, Ohio

Abstract - A decentralized variable structure control algorithm is developed for robotic manipulators to cause sliding modes to exist. New approaches are used to reach the sliding mode and to reduce non-ideal characteristics while in the sliding mode. Both position control and trajectory following are examined through two-link and three-link digital simulations.

I. INTRODUCTION

Variable structure system (VSS) theory has been applied to the control of manipulators in recent years [1], [3]. The major advantage of this technique is its insensitivity to parameter variations and disturbances once in the sliding mode. Therefore, exact modeling and load forecasting are not required. Also, the inherent nonlinear interactions between links can be rejected as disturbances. If such disturbances can be measured, the sliding mode is fairly insensitive to measurement error [2].

A major consideration in the design of VSS is the guarantee that the switching surface will be reached. The approach to be used satisfies the reaching condition with a very predictable state trajectory. The time required to reach the switching surface is given explicitly and is then used to reduce chattering while in the sliding mode.

II. CONTROL STRATEGY

Consider the system

$$\dot{x}_1 = x_2 \ , \tag{1a}$$

$$\dot{x}_2 = u \ , \tag{1b}$$

for which a sliding mode control is to be designed. The switching line is defined by s=0, where

$$s = cx_1 + x_2 \ , \ c>0 \ , \tag{2}$$

and the general reaching condition is

$$\dot{s} \cdot s < 0 \tag{3}$$

The objective of the new control algorithm is to keep $\dot{s}$ equal to a constant P, with the sign necessary to satisfy (3), that is

$$\dot{s} = -P \, \mathrm{sgn}(s) \ , \ P>0 \ . \tag{4}$$

If this is achieved, (3) is satisfied and the switching surface will be reached. Note that the inequality in (3) allows considerable variance in the regulation of $\dot{s}$ without violation of the reaching condition.

Given (1) and (4), the solutions for x_1 and x_2 can be determined exactly for the reaching phase of motion. They are

$$x_1(t) = -\frac{P}{c}t\,\mathrm{sgn}(s) - \frac{1}{c}\left(\frac{P}{c}\mathrm{sgn}(s)+x_2(0)\right)e^{-ct} + \left[\frac{1}{c}\left(\frac{P}{c}\mathrm{sgn}(s)+x_2(0)\right)+x_1(0)\right] , \quad (5)$$

$$x_2(t) = -\frac{P}{c}\,\mathrm{sgn}(s) + \left(\frac{P}{c}\,\mathrm{sgn}(s) + x_2(0)\right)e^{-ct} . \quad (6)$$

The time required to reach the switching line is found by solving (2), (5), and (6). The resulting simple relationship is

$$t_1 = |cx_1(0) + x_2(0)|/P , \text{ or} \quad (7a)$$

$$t_1 = |s(0)|/P . \quad (7b)$$

This shows that increasing P, and therefore $\dot{s}$, directly reduces the time to reach the switching line. A shorter reaching time is desirable, since once the trajectory lies on the switching line, the system solution depends only on the design parameter c and the sliding mode initial conditions. The solution is

$$x_1(t) = x_1(t_1)e^{-c(t-t_1)} , \quad t \geq t_1 . \quad (8)$$

The time at which the system reaches within some ϵ of the origin is

$$t_2 = \frac{1}{c}\ln\left|\frac{x_1(t_1)}{\epsilon}\right| + t_1 . \quad (9)$$

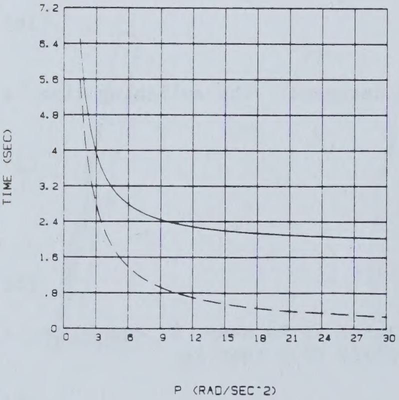


Fig. 1. Periods of motions, t_1 (dashed) and t_2 (solid).

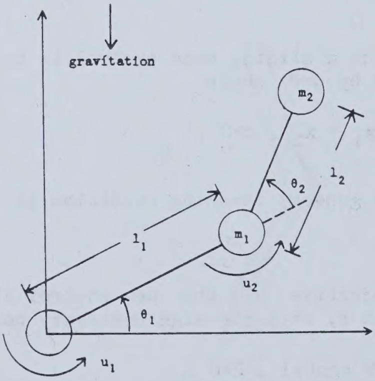


Fig. 2. A two-link manipulator.

Figure 1 shows t_1 and t_2 as a function of P. In the design process, P should be chosen large to minimize the reaching time. However, figure 1 shows that although t_1 continues to decrease with increasing P, the total time of the trajectory, t_2, decreases much more slowly for high values of P. The motivation to limit P is that the control necessary to regulate $\dot{s}$ increases as P increases. This will become more apparent in the next section.

III. REDUCTION OF NON-IDEAL CHARACTERISTICS

Non-ideal characteristics of the sliding mode such as chattering result primarily from switching delays. The same control law which brings the system quickly to the switching line will drive the system considerably past the switching line in one switching period, T_s . That is, from a position near the switching line, t_1 is much smaller than the switching period. To reduce chattering, t_1 from $|s|$ less than some Δ must be increased to a value greater than $T_s/2$. This is accomplished by reducing P, and therefore $\dot{s}$.

A self-adapting algorithm to adjust P is used to maintain insensitivity in chatter reduction to parameter variations and disturbances. If the trajectory crosses the switching line and exceeds the specified Δ within one sampling period, P is decreased by a factor of 0.625. If, however, the trajectory exceeds the limit without having crossed the switching line in the previous sampling period, P is increased by a factor of 2.0. P is allowed to increase faster than decrease to respond rapidly to sudden changes in the loading of the manipulator.

IV. TWO-LINK SIMULATION

The manipulator used is presented in [1]. The model and angles are defined in figure 2. Additional inertias about the center of gravity, J_1 and J_2, are also included. The full dynamic equations are used in the simulation and are given below [1].

$$\begin{vmatrix} \alpha_{11}(\theta_2) & \alpha_{12}(\theta_2) \\ \alpha_{12}(\theta_2) & \alpha_{22} \end{vmatrix} \begin{vmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{vmatrix} = \begin{vmatrix} \beta_{12}(\theta_2)\dot{\theta}_1^2 + \beta_{12}(\theta_2)\dot{\theta}_1\dot{\theta}_2 \\ -\beta_{12}(\theta_2)\dot{\theta}_2^2 \end{vmatrix} + \begin{vmatrix} \gamma_1 g \\ \gamma_2 g \end{vmatrix} + \begin{vmatrix} u_1 \\ u_2 \end{vmatrix} \tag{10}$$

g is the gravitational constant and u_j are the applied torques.

$$\alpha_{11}(\theta_2) = (m_1+m_2)l_1^2 + m_2 l_2^2 + 2m_2 l_1 \cos(\theta_2) + J_1 \tag{11}$$

$$\alpha_{22} = m_2 l_2^2 + J_2 \tag{12}$$

$$\alpha_{12}(\theta_2) = m_2 l_2^2 + m_2 l_1 l_2 \cos(\theta_2) + m_2 l_2 \cos(\theta_1+\theta_2) \tag{13}$$

$$\beta_{12}(\theta_2) = m_2 l_1 l_2 \sin(\theta_2) \tag{14}$$

$$\gamma_1(\theta_1,\theta_2) = -((m_1+m_2) l_1 \cos(\theta_2) + m_2 l_2 \cos(\theta_1+\theta_2)) \tag{15}$$

$$\gamma_2(\theta_1,\theta_2) = -m_2 l_2 \cos(\theta_1+\theta_2) \tag{16}$$

The values used in the simulation experiments are $m_1 = 0.5$ kg, $m_2 = 6.25$ kg, $l_1 = 1.0$ m, $l_2 = 0.8$ m, $J_1 = 5$ kg m^2, and $J_2 = 5$ kg m^2.

As suggested in [2], interactions between links are considered as disturbances and rejected by the sliding mode to provide decoupling. The program flow of calculations including sampling instants is illustrated in figure 3. The detailed equations are given below.

The simplified model is given by

$$T_j = (m_j l_j^2 + J_j)\ddot{\theta}_j + f_j \text{ , } j=1,2, \tag{17}$$

where f_j is the disturbance. The disturbance is calculated from

$$f_j = T_j - (m_j l_j^2 + J_j)\ddot{\theta}_j \text{ .} \tag{18}$$

The disturbance measurement is filtered digitally to remove the components of the order of the switching frequency. This removes the inherent noise of numerical differentiation and retains the lower frequency actual disturbance value. The interaction disturbances which depend on the position of the manipulator and the velocities of other links are of lower frequencies. The disturbances due to the acceleration of other links change at the switching frequency. The average of these disturbances depends on the equivalent control [1] which is, however, a lower frequency signal.

For position control, the angular error and velocity of each link are the system states. The switching lines are defined by

$$s_j = c_j e_j + \dot{\theta}_j = 0 \text{ , and} \tag{19}$$

$$e_j = \theta_j - \theta_{jd} \text{ , } j=1,2. \tag{20}$$

The desired accelerations are found by taking the time derivative of (12) and applying the control strategy (4).

$$\dot{s}_j = c_j \dot{\theta}_j + \ddot{\theta}_{jd} = -P \operatorname{sgn}(s) \text{ , } j=1,2. \tag{21}$$

Solving for the accelerations,

$$\ddot{\theta}_{jd} = -c_j \dot{\theta}_j - P \operatorname{sgn}(s) \text{ .} \tag{22}$$

The control torque is calculated from (17) and (22),

$$T_{jd} = (m_j l_j^2 + J_j)\ddot{\theta}_{jd} + f_j \text{ .} \tag{23}$$

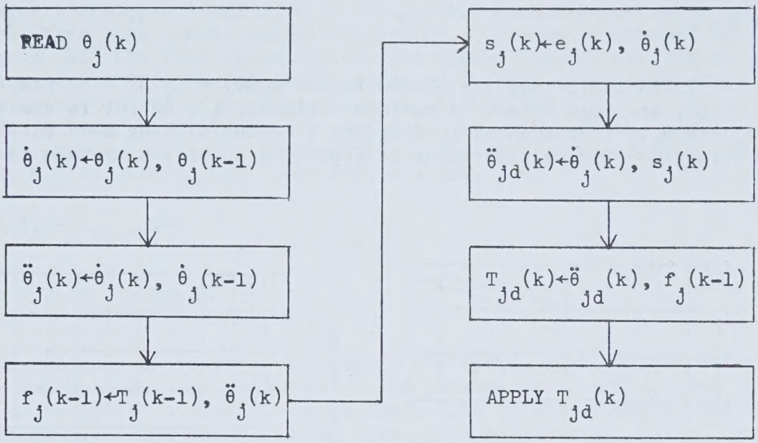

READ $\theta_j(k)$
$\dot{\theta}_j(k) \leftarrow \theta_j(k),\ \theta_j(k-1)$
$\ddot{\theta}_j(k) \leftarrow \dot{\theta}_j(k),\ \dot{\theta}_j(k-1)$
$f_j(k-1) \leftarrow T_j(k-1),\ \ddot{\theta}_j(k)$
$s_j(k) \leftarrow e_j(k),\ \dot{\theta}_j(k)$
$\ddot{\theta}_{jd}(k) \leftarrow \dot{\theta}_j(k),\ s_j(k)$
$T_{jd}(k) \leftarrow \ddot{\theta}_{jd}(k),\ f_j(k-1)$
APPLY $T_{jd}(k)$

Fig. 3. Program flow of calculations.

Although no actuator dynamics are included in this simulation, one must place practical limits on any control input. The maximum torques applied in the control of the same system in [1] were used as these limits. As long as the desired torque is obtainable, the desired acceleration is produced, and (21) is satisfied. Then the switching line will be reached as previously calculated. Note also that the measured disturbances are rejected even during the reaching phase of motion if this is the case. If the disturbance is too large, causing the torque limit to be exceeded, $\dot{s}$ will not remain constant. However, (3) may still be satisfied so that the switching line is reached. In general the switching line will be reached if a large enough controller gain is used [5]. A major component of the disturbance is due to the accelerations of other links. (22) shows that the magnitude of P affects these disturbances directly . Lower values of P keep overall control and thus disturbances lower and allow complete disturbance rejection and good regulation of $\dot{s}$.

The simulation experiments show that the manipulator joint angles are regulated to their desired positions. The reaching phase also is as expected. $\dot{s}$ is kept near its setpoint and the reaching times are close to those predicted by (7).

The example discussed in [1] will be considered here, also. The manipulator starts at rest with non-zero initial angles and moves to the zero angle position of each link. Specifically, with

$$x_1 = \theta_1 - \theta_{1d} , \tag{24}$$

$$x_2 = \dot{\theta}_1 , \tag{25}$$

$$x_3 = \theta_2 - \theta_{2d} , \tag{26}$$

$$x_4 = \dot{\theta}_2 , \tag{27}$$

$x_1(0) = -2.784$ rad., $x_2(0) = -1.204$ rad., $x_2(0) = x_4(0) = 0$ rad./sec., $\theta 1_d = 0$ rad., and $\theta 3_d = 0$ rad. A maximum variance, Δ , of .01 is chosen for each s_1 and s_2. P is adapted to decrease $\dot{s}$ during sliding mode to satisfy this criteria. Torque chattering is kept to a specified minimum in this way.

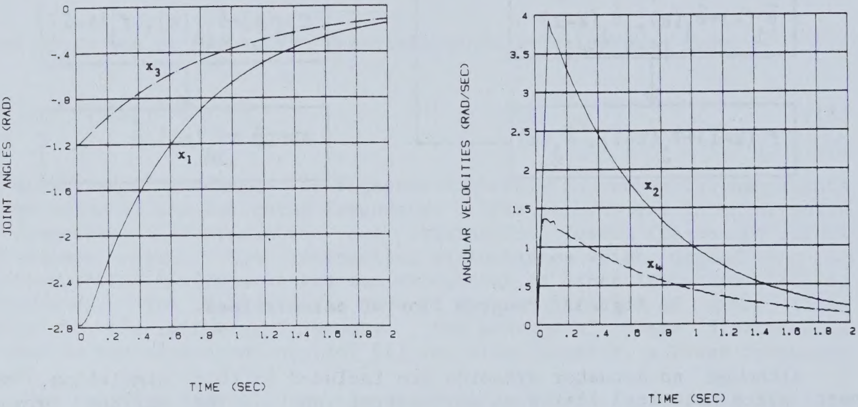


Fig. 4. System states during a position control experiment.

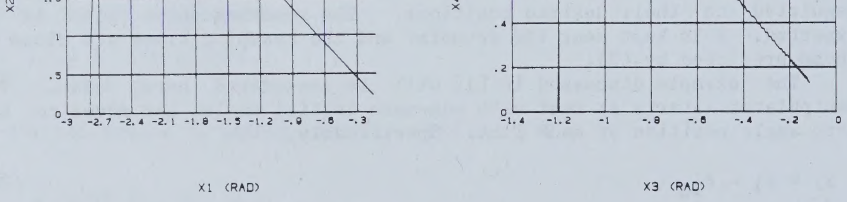


Fig. 5. Phase plane trajectories for a position control experiment.

The set of switching line parameters for consideration are $c_1 = 1.5$, $c_2 = 1.2$, three times faster than in [1]. Sliding mode time constants from $c_1 = .5$ to $c_1 = 6$ and $c_2 = .4$ to $c_2 = 4.8$ were examined with similar results from the same control algorithm. Figure 4 shows the time responses of the four system states and figure 5 shows the phase plane trajectories. Once the trajectories reach the switching line, the system slides to the origin in a close-to-ideal sliding mode.

Trajectory control is accomplished by using both position and velocity error states. The desired trajectory is specified by $\theta_{1d}(t)$, $\dot{\theta}_{1d}(t)$, $\theta_{2d}(t)$, and $\dot{\theta}_{2d}(t)$. The system states are

$$e_j = \theta_j - \theta_{jd} \text{ , and} \tag{28}$$

$$w_j = \dot{\theta}_j - \dot{\theta}_{jd} \text{ , } j=1,2. \tag{29}$$

In general, to follow a trajectory the system must respond quickly to error conditions. For this reason the parameters c_1 and c_2 are chosen large to bring the system states to the origin quickly. Due to the moving trajectories, the error states are not necessarily going to move consistently toward the origin as with position control. However, if the Δ 's are chosen smaller the system will stay closer to the switching line and will follow the trajectory more closely.

A typical trajectory is straight-line motion of the end of the manipulator. The beginning and ending Cartesian coordinates of the trajectory to be considered are (.8, 1.5) and (1.77, .3), respectively. Linear acceleration and deceleration periods of .25 sec. are included in the 1.8 sec. move. Constant linear velocity is maintained during the center portion of the move. The reverse kinematic equations which produce the joint angles for this trajectory are derived as in [4] and are given in [6]. Figure 6 shows the joint angles during the move. The desired angles for the trajectory are not plotted because the error is not visible in this figure.

Although the state trajectories are much less predictable in trajectory control, torque chattering around the switching line can still be observed. Since the state remains near the switching line, the system moves more toward the origin than if the sliding mode had not been introduced. In general, the joint angle and angular velocity errors are maintained with opposite signs which is necessary for the system to move toward the origin.

V. THREE-LINK SIMULATION

The dynamics of the three links of the Rhino XR-1 closest to the base have been derived [6]. The model used, including the joint angles and model parameters, is shown in figure 7. Lumped equivalent masses are assumed and an additional moment of inertia, J_1, is included only for the base of the manipulator. The full dynamic equations are used in the simulation and are given below.

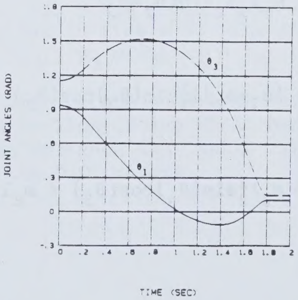


Fig. 6. Joint angles during a trajectory experiment.

$$\begin{vmatrix} \alpha_{11}(\theta_2,\theta_3) & 0 & 0 \\ 0 & \alpha_{22} & \alpha_{23}(\theta_2,\theta_3) \\ 0 & \alpha_{23}(\theta_2,\theta_3) & \alpha_{33} \end{vmatrix} \begin{vmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 \end{vmatrix} =$$

$$\begin{vmatrix} \beta_{11}(\theta_2,\theta_3)\dot{\theta}_1\dot{\theta}_2 + \beta_{12}(\theta_2,\theta_3)\dot{\theta}_1\dot{\theta}_3 \\ -\beta_{21}(\theta_2,\theta_3)\dot{\theta}_3^2 - \beta_{22}(\theta_2,\theta_3)\dot{\theta}_1^2 \\ \beta_{21}(\theta_2,\theta_3)\dot{\theta}_2^2 - \beta_{32}(\theta_2,\theta_3)\dot{\theta}_1^2 \end{vmatrix} + \begin{vmatrix} 0 \\ \gamma_2(\theta_2,\theta_3)g \\ \gamma_3(\theta_3)g \end{vmatrix} + \begin{vmatrix} T_1 \\ T_2 \\ T_3 \end{vmatrix} \tag{30}$$

$$\alpha_{11}(\theta_2,\theta_3) = (m_1+m_2)l_1^2\cos^2(\theta_2)+m_2l_1l_2\cos(\theta_2)\cos(\theta_3)+m_2l_2^2\cos^2(\theta_3)+J_1 \tag{31}$$

$$\alpha_{22} = (m_1+m_2)l_1^2 \tag{32}$$

$$\alpha_{23}(\theta_2,\theta_3) = m_2l_1l_2\sin(\theta_2 + \theta_3) \tag{33}$$

$$\alpha_{33} = m_2l_2^2 \tag{34}$$

$$\beta_{11}(\theta_2,\theta_3) = 2(m_1+m_2)l_1^2\sin(\theta_2)\cos(\theta_2) + m_2l_1l_2\sin(\theta_2)\cos(\theta_3) \tag{35}$$

$$\beta_{12}(\theta_2,\theta_3) = m_2l_1l_2\cos(\theta_2)\sin(\theta_3) + 2m_2l_2^2\sin(\theta_3)\cos(\theta_3) \tag{36}$$

$$\beta_{21}(\theta_2,\theta_3) = m_2l_1l_2\sin(\theta_2-\theta_3) \tag{37}$$

$$\beta_{22}(\theta_2,\theta_3) = (m_1+m_2)l_1^2\sin(\theta_2)\cos(\theta_2) +m_2l_1l_2\sin(\theta_2)\cos(\theta_3) \tag{38}$$

$$\beta_{32}(\theta_2,\theta_3) = m_2l_2^2\sin(\theta_3)\cos(\theta_3) + m_2l_1l_2\cos(\theta_2)\sin(\theta_3) \tag{39}$$

$$\gamma_2 = -(m_1+m_2)l_1\cos(\theta_2) \tag{40}$$

$$\gamma_3 = -m_2 l_2 \cos(\theta_3) \tag{41}$$

The values of the parameters used are $J_1 = 0.0243$ kg m^2, $m_1 = 0.624$ kg, $m_2 = 0.782$ kg, and $l_1 = l_2 = .23$ m. The actuator dynamics were also included in this simulation. The model used for the D.C. motors is described by

$$\dot{x} = -(R/L)x + v - K_v\omega \tag{42}$$

$$T = (K_T/L)x \tag{43}$$

where v is the applied voltage, ω is the motor speed, and T is the output torque. The values of the parameters used are R = 2.04 Ω, L=1.49 mH, $K_v = 0.0194$ volts/rad/sec, and $K_T = 19.4$ mN/mA. The maximum control input used was 12 volts.

Including the actuator dynamics in the simulation introduces the practical difficulty of not knowing the actual torques applied to the manipulator. Consequently, a somewhat different approach is used. The disturbance measurement is updated by the acceleration error from the preceding sampling period. That is

$$f_j(k-1) = J_j(\ddot{\theta}_{jd}(k-1) - \ddot{\theta}_j(k)) + f_j(k-2)\ . \tag{44}$$

Then a simplified model of the D.C. motor is used to transfer the desired torque to the control variable. The model is given by

$$T_j = (K_T/R)(v_j - K_v\omega_j)\ , \tag{45}$$

and the desired voltage is given by

$$v_{jd} = (R/K_T)T_{jd} + K_v\omega_j\ . \tag{46}$$

Any error introduced by this simplication will appear in the calculations like a disturbance and, therefore, is rejected by the algorithm as a disturbance. Thus, no decrease in performance should result and none was observed.

Position control experiments were run with good results. The initial conditions were $\theta_1 = 1.4$ rad., $\theta_2 = 1.05$ rad., $\theta_3 = -0.52$ rad., and $\dot{\theta}_1 = \dot{\theta}_2 = \dot{\theta}_3 = 0$. The final position was $\theta_{1d} = \theta_{2d} = \theta_{3d} = 0$ rad. The phase plane trajectories of the test are given in figure 8. The states used are $x_1 = \theta_1 - \theta_{1d}$, $x_2 = \dot{\theta}_1$, $x_4 = \theta_2 - \theta_{2d}$, $x_5 = \dot{\theta}_2$, $x_7 = \theta_3 - \theta_{3d}$, and $x_8 = \dot{\theta}_3$.

The final phase of this study will be the real-time application of this control algorithm to the three-link Rhino XR-1 hardware. Pulse width modulation hardware has been designed and built in the O.S.U. Digital Control Laboratory to drive the D.C. motors.

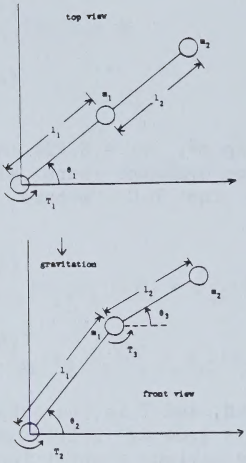


Fig. 7. A three-link manipulator.

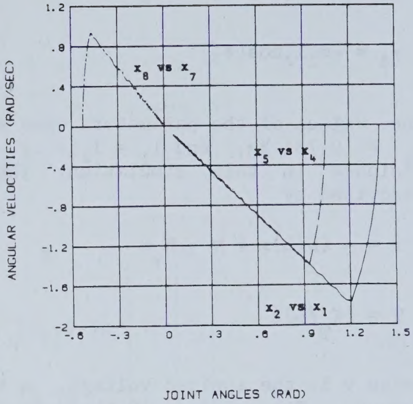


Fig. 8. Three-link simulation results.

VI. CONCLUSION

The simulation experiments have demonstrated the effectiveness of the control strategy outlined in section II. The time to reach the switching line was derived and then used to develop an adaptive algorithm to reduce torque chattering. The same control was applied to both position and trajectory control with good results. As is characteristic of systems under sliding mode control, nonlinear interactions and disturbances were rejected without complex compensation algorithms. This material is based upon work partially supported by the National Science Foundation under Grant No. ECS-8203877.

VII. REFERENCES

[1] K.-K. D. Young, "Controller design for a Manipulator using theory of variable structure systems," IEEE Trans. Syst., Man, Cybern., vol. sm C-8, pp. 101-109, Feb. 1978.

[2] V. I. Utkin, "Variable structure systems with sliding modes: a survey," IEEE Trans. Automat. Contr., vol. AC-22, pp. 212-222, Apr. 1977.

[3] C. A. Klein and J. J. Maney, "Real-time control of a multiple-element mechanical linkage with a microcomputer," IEEE Trans. Ind. Electron. Contr. Instrum., vol. IECI-26, pp. 227-234, Nov. 1979.

[4] R. P. Paul, Robotic Manipulators: Mathematics, Programming, and Control. Cambridge, MA: MIT Press, 1981.

[5] U. Itkis, Control systems of variable structure, NY, NY: Halsted Press, 1976.

[6] R. G. Morgan, "Variable Structure Control Sytems for Robotic Manipulators", in preparation.

FEEDBACK OBSTACLE AVOIDANCE CONTROL

BRUCE H. KROGH
Carnegie-Mellon University
Department of Electrical and Computer Engineering
Schenley Park
Pittsburgh, Pennsylvania 15213

ABSTRACT

An important problem in the control of robotic manipulators and unmanned vehicles is to determine trajectories which transfer the system from an initial point to a final goal while avoiding collisions with obstacles in the workspace. Necessary conditions for such trajectories derived from control theoretic formulations of the problem do not provide solutions for practical applications. The control is normally computed to follow a path which has been pre-determined to be acceptable. This paper presents a feedback obstacle avoidance control which navigates a point among obstacles in the plane. The control at each instant is computed from sensor information and takes into account the nominal system dynamics. It is applicable to both real-time control and off-line path planning.

INTRODUCTION

An avoidance control is an admissible control which steers the state of a system so that it avoids a specified subset of the state space. Various avoidance control problems have been formulated in the control theory literature: references [1-3] concern the existence of an avoidance control where the sole objective is to remain outside the avoidance set $\mathcal{A}$; in [4-5] avoidance control is considered in the context of pursuit-evasion games where the system is controlled by two agents; the avoidance control of [6] is required to transfer the system from a given initial state to a given terminal state while maximizing the minimum distance between the state trajectory and $\mathcal{A}$. This latter case is most similar to the obstacle avoidance control problem considered in the present paper. The results of [1-6] are of limited applicability since $\mathcal{A}^c$ is normally required to be convex and the resulting necessary and sufficient conditions do not immediately provide for construction of the avoidance control in non-trivial examples.

The present work is motivated by the problem of obstacle avoidance in guidance of unmanned vehicles and control of robotic manipulators. In these applications the Obstacle Control Avoidance (OAC) is normally determined in three steps. First, a collision-free path is found among the obstacles in the workspace [7-9]. Second, the nominal dynamic constraints are taken into account in computing a velocity trajectory along the path [10-12]. Finally, this nominal trajectory is tracked by servo-controllers [13, 14]. We refer to the first two steps as the supervisory control of the system.

Methods proposed for step one, computing the collision-free path, normally assume knowledge of the entire space to be navigated including the locations and shapes of all obstacles. The path finding algorithms are too complex for real-time applications. Moreover, since dynamics are ignored the computed paths are often unrealistic or inefficient and must therefore be smoothed and refined in step two. The second step produces a path which closely follows the result from step one while accounting for the velocity and acceleration capabilities of the system. These dynamic constraints may be incorporated into an optimal formulation, minimizing time or energy for example, however, the nominal path is fixed by step one. Thus, the nominal OAC is determined in an open-loop fashion.

In constrast to the above approach, the algorithm proposed in the sequel is a feedback scheme for supervisory control which produces an OAC satisfying nominal acceleration constraints. The control is computed at each instant based upon information which would be available from real-time sensing. The computational simplicity of the algorithm makes it appropriate for online applications. Potential application for off-line trajectory planning is also discussed in the concluding section. The feedback control scheme is demonstrated for the particular case of navigating a point among obstacles in the plane. This is a first step towards the development of more general supervisory control schemes for robotic systems which have a feedback rather than open-loop structure.

The paper is organized as follows. The problem in formulated and notation is introduced in Section 2. Section 3 presents the feedback algorithm and section 4 describes a simulation program with some examples. The concluding section suggests several directions for further research.

PROBLEM FORMULATION

As a prototypic supervisory control problem we consider obstacle avoidance in the plane. This has been a common starting point for the development of open-loop path-finding algorithms [7-9]. To develop and demonstrate the feedback scheme we characterize the trajectory of the system by

$$\begin{bmatrix} x(t+1) \\ v(t+1) \end{bmatrix} = \begin{bmatrix} I & \Delta I \\ 0 & I \end{bmatrix} \begin{bmatrix} x(t) \\ v(t) \end{bmatrix} + \begin{bmatrix} \frac{\Delta^2 I}{2} \\ \Delta I \end{bmatrix} u(t) \qquad (1)$$

where $x(t) \in R^2$ is the position in Cartesian coordinates and $v(t) \in R^2$ is the velocity. Equation (1) represents the sampled-data dynamics with sampling period Δ. The magnitude of the acceleration control, $u(t) \in R^2$, is constrained as

$$||u(t)|| \leq \bar{a} \qquad (2)$$

The avoidance set is the union of N obstacle sets in the plane, $A = \bigcup_{i=1}^{N} A_i$, where each obstacle set (obstacle) A_i is closed and convex. For our purposes the controlled system is considered to be the point x(t) since the physical obstacles can be represented by larger sets which account for the dimensions of the controlled system [7].

The objective of the OAC is to transfer the system from its initial position, $x(0) = x^0$, to the goal position, $x(T) = x^g$, where the initial and final velocities are zero (at rest). The final time, T, is unspecified. The trajectory must avoid the obstacles, that is, $x(t) \notin A$ for all $0 \leq t \leq T$.

We assume that at each instant the system sensors provide the following information:

- Direction and distance to each obstacle set visible from x(t).

- Direction and distance to the visible edges of at least one selected obstacle

where "visible" means the line-of-sight from x(t) to the obstacle, or obstacle edge.

The following notation and definitions will be used in the sequel to represent the available sensory information.

- The point on obstacle A_i closest to x(t):

$$y_i(t) \triangleq \arg \{\min_{y \varepsilon A_i} ||x(t)-y_i||\} \tag{3}$$

- The distance from x(t) to obstacle A_i:

$$\delta_i(t) \triangleq ||x(t) - y_i(t)|| \tag{4}$$

- The unit vector directed from x(t) to A_i:

$$d_i(t) \triangleq 1/\delta_i(t) \quad (y_i(t) - x(t)) \tag{5}$$

- An edge of obstacle A_i will be denoted $z_i(t)$.
- The distance to edge $z_i(t)$ of obstacle A_i.

$$\varepsilon_i \triangleq ||x(t) - z_i(t)|| \tag{6}$$

- The direction to edge $z_i(t)$ of obstacle A_i

$$e_i \triangleq 1/\varepsilon_i \quad (x(t) - z_i(t)) \tag{7}$$

The notion of an edge of an obstacle has been defined and investigated in the literature on computer vision [15]. For our purposes it is assumed the sensors provide the required edge information.

The control is to be implemented in feedback, that is, an OAC is to be computed at each instant t=0, 1,... using the available sensory information at time t which will be denoted by I_t.

THE FEEDBACK ALGORITHM

The proposed Feedback Obstacle Avoidance Control (FOAC) belongs to the class of heuristic control schemes minimizing an instantaneous criterion. At each instant the system is accelerated in the "best" direction so as to navigate toward x^g while avoiding collisions with obstacles A_i. At each instant t, the control is determined as the solution to a simple optimization problem defined in terms of an objective vector $c(I_t) \varepsilon R^2$,

$$\max c(I_t)' u \tag{8}$$

subject to the control magnitude constraint (2). The solution to (8) results in acceleration of magnitude $\bar{a}$ in the direction of the cost objective vector, that is,

$$u(I_t) = \bar{a} \quad \frac{c(I_t)}{||c(I_t)||} \tag{9}$$

Thus, the feedback scheme relies on an appropriate selection of $c(I_t)$ at each instant.

The feedback loop is closed by the computation of $c(I_t)$ which is decomposed as

$$c(I_t) = c_0(I_t) + \alpha c_g(I_t) \tag{10}$$

where

$c_0(I_t)$: Obstacle avoidance vector

$c_g(I_t)$: Goal attraction vector

α : Goal attraction weighting

If no obstacles are present, $c_0(I_t) = 0$, and the computation of $c(I_t) = c_g(I_t)$ leads to a time-optimal control of the system from x^0 to $x(T) = x^g$ with $v(0) = v(T) = 0$. When obstacles are present, $c_0(I_t)$ is the weighted sum of vectors directed away from the obstacles, that is,

$$c_0(I_t) = -\sum_{i=1}^{N} \beta_i d_i(t) \quad . \tag{11}$$

The obstacle feedback penalty, β_i, is computed from I_t to represent the relative urgency of accelerating in direction $-d_i(t)$ so as to avoid obstacle A_i. The obstacle feedback penalty accounts for the dynamic constraints by effectively comparing the control required to avoid the obstacle to the available acceleration capacity.

The presence of obstacles also affects the computation of $c_g(I_t)$. When an obstacle lies between $x(t)$ and x^g it is necessary to steer the system around the obstacle rather than directly to x^g. A selected edge of the obstacle becomes an intermediate goal and $c_g(I_t)$ is directed to result in acceleration around the obstacle. The detailed computations are described below.

The FOAC is feedback in philosophy and structure. The control at each instant reflects a "best" response to the immediate operating conditions rather than the initial step of an open-loop control computed over a time horizon. The instantaneous control $u(I_t)$ is appropriate for short-term acceleration, while the overall OAC objectives are realized by repeated computation of $c_0(I_t)$ and $c_g(I_t)$. The seeming myopic character of the instantaneous control is mitigated by the incorporation of new feedback information at each stage and by the manner in which the obstacle feedback penalties represent the system's dynamic constraints. This is demonstrated by the simulation results in the following section.

Goal Attraction Vector - No Obstacles Present

Controlling the system (1) from x^0 to x^g when $A = \phi$ can be solved by a simple time optimal control of magnitude $\pm\bar{a}$ in the direction of the goal. This is a single (spatial) dimension problem with the well-known bang-bang solution with a single switch time [16]. We note that for the single-spatial dimension problem with initial velocity v and distance d to go to the origin the minimum time for an acceleration constrainted control is

$$T(v,d) = \begin{cases} \dfrac{\sqrt{2(v^2 + 2\bar{a}\ d)} - v}{\bar{a}} & \text{if condition 1.} \\[2ex] \dfrac{\sqrt{2(v^2 - 2\bar{a}\ d)} + v}{\bar{a}} & \text{if condition 2.} \end{cases} \tag{12}$$

where

condition 1: $(v < 0 \text{ and } 2\bar{a}\ d \geq -v^2)$ or $(v \geq 0 \text{ and } -2\bar{a}d < -v^2)$

and

condition 2: $(v \leq 0 \text{ and } 2\bar{a}d < -v^2)$ or $(v > 0 \text{ and } -2\bar{a}d \geq -v^2)$

Also, the initial control value, $u(0^+)$ is given by

$$u(0^+) = \begin{cases} \bar{a} & \text{if} \quad \text{condition 1} \\ -\bar{a} & \text{if} \quad \text{condition 2} \end{cases}$$

For our problem we wish to compute $c_g(I_t)$ so that the resulting control (9) is in fact this time-optimal solution when $c_0(I_t) \equiv 0$. Toward that end, we define the distance to the goal,

$$\delta_1^g(t) = ||x(t) - x^g||, \tag{13}$$

and the unit vector pointing to the goal

$$g_1(t) = \frac{1}{\delta_1^g(t)} (x(t) - x^g) . \tag{14}$$

The velocity toward the goal, $v_1^g(t)$, is given by

$$v_1^g(t) \triangleq \langle g_1(t), v(t)\rangle , \tag{15}$$

where $\langle\ ,\ \rangle$ is the vector scalar product.

We weight the goal cost vector in the direction toward the goal $g_1(t)$ by the minimum time-to-go if the time optimal solution were implemented. Moreover, this weight is positive if the initial time-optimal control value would be $+\bar{a}g_1(t)$ negative if it would be $-\bar{a}g_1(t)$. As is demonstrated by the simulations of the following sections, this does indeed result in a time-optimal control when no obstacles are present.

If, after avoiding some obstacles, the goal is in sight, then the velocity of the system is not necessarily aligned with $g_1(t)$. To guarantee convergence of the trajectory to $x(T) = x^g$, $v(T) = 0$, a vector is added to $c_g(t)$ which is orthogonal to $g_1(t)$, that is, we define

$$g_2(t) \triangleq \text{unit vector ortogonal to } g_1(t) \tag{16}$$

and the velocity in direction $g_2(t)$,

$$v_2^g(t) \triangleq \langle g_2(t), v(t)\rangle \tag{17}$$

The time-optimal solution in this orthogonal direction determines the weighting of $c_g(t)$ in direction $g_2(t)$. Then we have, for the case when no obstacles are between $x(t)$ and x^g,

$$c_g(t) = \gamma_1(t)\, g_1(t) + \gamma_2(t)\, g_2(t) \tag{15}$$

where $g_1(t)$, $g_2(t)$ are defined by (14), and (16)

and

$$\gamma_i(t) = \begin{cases} T[v_i^g(t),\ \delta_i^g(t)] & \text{if condition 1} \\ -T[v\ (t),\ \delta_i^g(t)] & \text{if condition 2} \end{cases} \quad ,\ i = 1,2 \tag{16}$$

Note that $\delta_2^g \equiv 0$.

Obstacle Avoidance Vector - Obstacle Feedback Penalties

The obstacle avoidance vector, $c_0(I_t)$ (11), depends on the obstacle feedback penalties which are computed as follows.

The velocity at which the system is approaching the line tangential to obstacle A_i at point $y_i(t)$ is given by

$$v_i^0(t) = \langle v,\ d_i(t)\rangle \quad . \tag{17}$$

If $v_i^0(t)$ is not positive, the system is traveling away from A_i and the obstacle is ignored, that is,

$$\beta_i \equiv 0 \quad \text{if } v_i^0(t) \leq 0 \quad . \tag{18}$$

When $v_i^0(t) > 0$, β_i depends upon an assessment the urgency of accelerating so as to avoid A_i. To decrease v_i^0 to zero as quickly as possible the system can be accelerated in the direction $-d_i(t)$ at a rate $\bar{a}$. Thus, we define the minimum avoidance deceleration time, T_i, as

$$T_i \triangleq \frac{v_i^0(t)}{\bar{a}} \quad . \tag{19}$$

The maximum constant deceleration time, computed as the time at which the tangent line would be attained but not crossed if a constant acceleration is applied in the direction $-d_i(t)$, is given as

$$\tau_i \triangleq \frac{2\delta_i(t)}{v_i^0(t)} \tag{20}$$

The obstacle feedback penalty is defined as the inverse of the reserve deceleration time, $T_i - \tau_i$, that is,

$$\beta_i = (T_i - \tau_i)^{-1} = \frac{\bar{a}\ v^0_{i(t)}}{v^2_{i(t)} - 2\delta_i(t)\ \bar{a}} \qquad \text{if } v_i^0\ (t) > 0 \quad . \tag{21}$$

From these definitions it is evident that $\beta_i \uparrow \infty$ as the constant deceleration required to avoid crossing the tangent line approaches the maximum possible deceleration. Thus, as a particular obstacle becomes more "critical", the cost vector, $c_0(I_t)$, is directed so as to cause $u(I_t)$ to accelerate the system away from the obstacle. Note that the resulting control is conservative in that the obstacle feedback penalties are computed in terms of the line supporting the convex obstacle at the nearest point on the obstacle, $y_i(t)$.

Goal Attraction Vector - Obstacles Present

When an obstacle is present between $x(t)$ and x^g it is necessary to compute $c_g(I_t)$ so as to direct the system around the obstacle while continuing to approach x^g as quickly as possible. Restricting the computation to include only the sensor information delineated above, it is necessary to introduce a rule for selecting the direction around the obstacle. One such rule, implemented in the simulations below, is to steer the system around the edge of the obstacle which is closest to the goal, x^g. To accomplish this, we decompose $c_g(I_t)$ into two components as in (15):

$$c_g(I_t) = \gamma_1(I_t) g_1(I_t) + \gamma_2(I_t)\, g_2(I_t) \tag{22}$$

where $g_1(t)$ is a unit vector directed towards x^g as defined in (14). The weight $\gamma_1(I_t)$ is computed as in (16) using the distance to the obstacle in the direction $g_1(t)$ rather than the total distance to x^g. Thus, $\gamma_1(I_t) g_1(t)$ directs the acceleration as the time-optimal approach to the obstacle in the direction of x^g.

The direction $g_2(t)$ is selected to be parallel to the line connecting the point $\bar{x}^0$ (the point on the obstacle in direction $g_1(t)$ from $x(t)$) and the edge $z_i(t)$ determined to be closest to x^g, as described above. The distance $\delta_2^g(I_t)$ is defined as the distance to the line connecting x^g and z_i in direction $g_2(t)$. The weight $\gamma_2(I_t)$ is computed from the time optimal weightings for direction $g_2(I_t)$. One further complication arises if yet another obstacle obstructs the line of sight from $x(t)$ to $z_i(t)$, in which case $\gamma_2(I_t)$ is set to zero.

SIMULATIONS

To test the effectiveness of the algorithm a computer program was written which permits the arbitrary placement of rectangular obstacles on a graphics display along with selected start and goal points. The sampling period Δ, the goal weight factor α (10), and the acceleration constraint $\bar{a}$ are entered as variables. The results of some representative experiments are shown in Figures 1-3. In these simulations the per unit values of the parameters were $\Delta = .1$, $\bar{a} = 2$, and the points were plotted along the trajectory at every 10Δ increment.

Figure 1 illustrates the time-optimal trajectory resulting from $c_g(I_t)$ when no obstacles are present. When obstacles are introduced, the FOAC steers the system around the edges of the obstacles according to the edge selection rule discussed above (Fig. 2). To demonstrate the affect of the goal weight factor, the value of α was 0.05 and 0.5 for the simulations in figures 2 and 3, respectively. Note that in the latter case the trajectory comes closer to the obstacles and also takes longer to reach x^g.

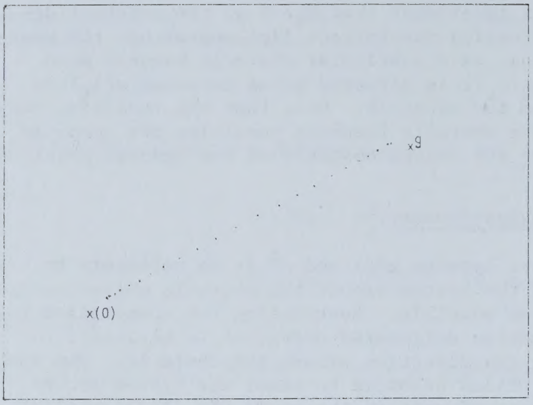


Figure 1. No obstacles present. FOAC results in time optimal solution, T = 24.0.

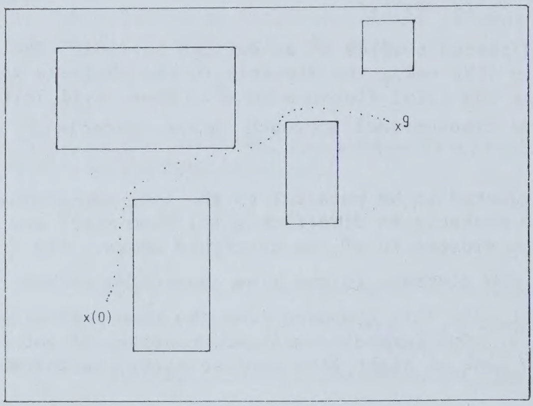


Figure 2. Obstacles present, goal weight, α = 0.05; T = 41.9.

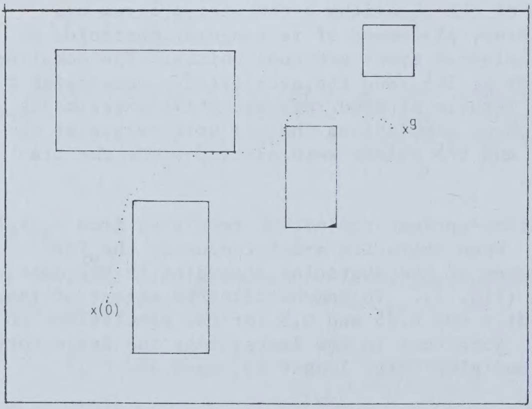


Figure 3. Obstacles present, goal weight, α = 0.5; T = 47.8.

DISCUSSION

In contrast to the usual open-loop approach to obstacle avoidance control the FOAC developed and demonstrated above offers the following advantages:

- The computations do not assume knowledge of the entire workspace.
- The trajectory is defined in terms of the acceleration of the system which is directly related to the actual controls (torque, force).
- The required computation is appropriate for real-time on-line control.
- Multiple off-line trajectory studies can be generated efficiently.

The last point suggests the application of the FOAC to off-line studies where various control strategies can be tested. The goal weighting factor a can be varied to generate trajectories of varying degrees of conservativeness. Also, different rules can be implemented for steering the system around obstacles.

The simulation testing of the algorithm demonstrates its effectiveness and suggests that several issues merit further investigation. For the relatively simple problem of obstacle avoidance in the plane, the dynamics of the feedback system should be studied analytically. The construction of the obstacle feedback penalties is intended to guarantee a collision-free path, however, this remains to be proved. It is also desirable to establish sufficient conditions which guarantee the FOAC provides an overall solution. The algorithm should be extended to the case of non-convex obstacles to guarantee trajectories which do not get "stuck" in corners.

The basic feedback structure demonstrated above may have potential as a general approach to the supervisory control of robotic systems. Such systems are characterized by a variety of sensors and actuators which must be coordinated to accomplish complex tasks. Presently, this coordination is accomplished in an open-loop fashion similar to the method described in the introduction for obstacle avoidance. On the supervisory control level the sensory information is often used only to determine whether or not the task is being accomplished as planned, that is, the measurements serve to monitor rather than control the system. A goal for future research is to extend the approach developed in this paper for feedback supervisory control to control and coordinate actuators and sensors to respond at each instant according to a quantitative assessment of the immediate operating conditions. This requires techniques for effectively characterizing the future evolution of the system state trajectory with respect to the operating constraints.

REFERENCES

[1] R. Aggarwal and G. Leitmann, "Avoidance Control", ASME J. Dynamic Syst. Meas. and Contr., pp. 152-154, June 1972.

[2] B.R. Barmish, W.E. Schmitendorf and G. Leitmann, "A Note on Avoidance Control", ASME J. Dynamic Syst., Meas. and Contr., pp. 69-70, March 1981.

[3] W.E. Schmiterdorf, B.R. Barmish, and B.S. Elenbogen, "Guaranteed Avoidance Control and Holding Control", ASME J. Dynamic Syst., Meas. and Contr. pp. 166-172, June 1982.

[4] G. Leitmann and J. Skowronski, "Avoidance Control", J. Optimiz. Theory Appl., vol. 23, pp. 581-591, Dec. 1977.

[5] G. Leitman, "Guaranteed Avoidance Feedback Control", IEEE Trans. Automat. Contr., vol. AC-25, pp. 850-851, August 1980.

[6] R. Aggarwal and G. Leitmann, "A Maximum Distance Problem", ASME J. Dynamic Syst., Meas. and Contr., pp. 155-158, June 1972.

[7] T. Lorano-Perez, and M.A. Wesley, "An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles", Commun ACM, vol. 22, No. 10, pp. 560-570, Oct. 1979.

[8] T. Lozano-Perez, "Automatic Planning of Manipulator Transfer Movements", IEEE Trans. Syst., Man, Cybern, vol. SMC-11, No. 10, Oct. 1981.

[9] R.A. Brooks, "Solving the Find-Path Problem by Good Representation of Free Space", IEEE Trans. Syst., Man, Cybern, vol. SMC-13, No. 3, Mar/Apr. 1983.

[10] J.Y.S. Luh and C.S. Lin, "Optimum Path Planning for Mechanical Manipulators", ASME J. Dynamic Syst., Meas. and Contr., pp. 142-151, June 1981.

[11] C.S. Lin, P.R. Chang and J.Y.S. Luh, "Formulation and Optimization of Cubic Polynomial Joint Trajectories for Mechanical Manipulators", Proc. 21st IEEE Conf. Decision Contr., Orlando, FL, Dec. 8-10, 1982, pp. 330-335.

[12] M. Vukobratovic and M. Kircanski, "A Method for Optimal Synthesis of Manipulation Robot Trajectories", ASME J. Dynamic Syst., Meas. and Contr., pp., 188-193, June 1982.

[13] C.P. Neuman and V.D. Tourassis, "Robot Control Issues and Insight", Proc. 3rd Yale Workshop Appl. Adaptive Syst. Theory, New Haven, CT June 15-17, 1983.

[14] A.J. Koivo and T.H. Guo, "Adaptive Linear Controller for Robotic Manipulators", IEEE Trans. Automat. Contr., vol. AC-28, No. 2, pp. 162-171, Feb. 1983.

[15] J.Y.S. Luh, "An Anatomy of Industrial Robots and Their Control", IEEE Trans. Automat. Contr., vol. AC-28, No. 2, pp. 133-153, Feb. 1983.

[16] M. Athans, and P.L. Falb, Optimal Control, McGraw-Hill, 1966.

A SINGULAR PERTURBATION APPROACH TO THE DYNAMIC CONTROL OF ELASTIC ROBOTS

ANTONIO FICOLA, RICCARDO MARINO, SALVATORE NICOSIA
Dipartimento di Ingegneria Elettronica - Seconda Università di Roma
Via O. Raimondo, I00173 - Roma (Italy).

ABSTRACT

The paper has an introductory character with reference to industrial robots in which elasicity at joints cannot be neglected.

Experencies done by simulation have proved that control algorithms obtained on the basis of knowledges on rigid models are not suitable for robots whose elasticity at joints is not negligible. In our present approach the possibility of studying elastic robots with the method of singular perturbations is investigated.

INTRODUCTION

The aim of the dynamic control of an industrial robot is to force its end point to follow a given trajectory in a suitable working space of N dimensions. This trajectory can be specified either off-line or on-line. When only a discrete number of points on the trajectory are to be reached at zero velocity - as in assembly robots - the control is said to be *point to point*. Otherwise, if the trajectory is continuously specified - as in tracking robots - we speak of *continuous path* control.

In this paper we refer to the robot model which is more suitable for designing continuous path control systems when the amount of elasticity taken into account is not negligible. More precisely we derive a model in which elasticity is acting as perturbation.

Experimental essays, confirmed by simulation, prove that in general control algorithms obtained under the assumption of perfect rigidity do not work for robot in which the elasticity at joints is not negligible. Indeed, some manipulators use special coaxial gear-boxes which are very light and compact, but present some torsion elasticity whose effects are still to be investigated.

Moreover, in a first approximation, distributed elasticity can be also considered. In fact one can think of adjusting the elastic parameters of the joint gear-boxes in order to model distributed elasticity.

Joint elasticity due to gear-boxes can be easily incorporated in the dynamical equations (usually Lagrange equations or Newton-Euler equations) which are used for the generation of the dynamical models of robot arms. By using special programs it is possible to generate symbolic dynamical models of robots either with rigid or elastic joints (1).

DYNAMIC MODELS

In order to model a robot with elastic joints, each body of the open chain which constitutes the mechanical part of the robot arm is split into two parts which are elastically coupled by the gear-box. The first part is rigidly connected to the rotor of the motor and is called the *driving body*; the second part is rigidly connected to the arm of the robot and is called the *driven body*.

Because of this splitting the model requires two times as many state variables as in the rigid case. Two more state variables are in fact needed for each body: the relative elastic displacement between the driving and the driven body and the corresponding velocity. The degrees of freedom in the elastic case are apparently double but from the point of view of actual motion capabilities of the robot arm remain the same as in the rigid case.

The dynamics of a N joints rigid robot is described by equations displaying the following structure

$$B(q)\,\ddot{q} = m(t) - C(q)\,\dot{\tilde{q}} + e(q) = m(t) + F(q,\dot{q}) \qquad (1)$$

where q is an N vector whose components represent the joint variables; $\dot{\tilde{q}}$ is a ν vector ($\nu = N(N+1)/2$) whose components are $\dot{q}_j.\dot{q}_k$ ($j = 1,\dots,N$; $k = j,\dots,N$); B(q) is the N$\times\nu$ matrix containing centrifugal and Coriolis terms; m(t) is an N vector of generalized forces; e(q) is an N vector containing terms due to variations of potential energy. N is the number of joints and of bodies constituting the robot arm; N gives exactly the degrees of freedom if the robot is not redundant.

It is worth mentioning that the components of the vector F(q,q) and the entries of the matrix B(q) are highly nonlinear. In fact it is not advisable to generate B(q) and $F(q,\dot{q})$ by hand: symbolic languages can actually be used to produce B(q) and $F(q,\dot{q})$ given the spatial geometry and the masses distribution of the robot (1).

The dynamic of a robot arm with elastic joints can be modeles by equations of the type (1) with the following peculiarities:
- as mentioned before for N elastic joints the number of second order equations is 2N, i.e. q is a 2N vector;
- B(q) is a 2Nx2N matrix in which one can separate the entries related to the driving bodies from the ones related to the driven bodies;
- the effect of elasticity are included in the term $F(q,\dot{q})$;
- the 2N vector m(t) displays zero components in correspondence to the acceleration of the driven bodies with respect to the driving ones; thus the control space is R^N.

Recalling that the matrix B(q) is nonsingular one can rewrite equations (1) in the case of elastuc joints as

$$\ddot{q} = B^{-1}(q)\,F(q,\dot{q}) + B^{-1}(q)\,m(t) \;\hat{=}\; f(q,\dot{q}) + B^{-1}(q)\,m(t) \qquad (2)$$

In the rigid case it is easy to show that the system (2) is completely controllable. In fact one can define a state feedback and a state dependent linear change of coordinates in the control space R^N so that the transformed system becomes a linear controllable system with controllability indices (2,...,2). It is well known that such a transformation does not affect the controllability property of the system and holds globally. More precisely let us define the new controls

$$v(t) = f(q(t),\dot{q}(t)) + B^{-1}(q(t))\, m(t) \tag{3}$$

The system (2) becomes

$$\ddot{q}(t) = v(t) \tag{4}$$

which is completely controllable.

Furthermore, by using the transformation (3), one can actually design control laws via linear system techniques (4).

The preceding argument is a simple application of a general theory on the linearization of nonlinear systems ((3),(4)) via feedback transformations.

We are currently investigating the applicability of those theorical results to the elastic model. It is our conjecture that the elastic model does not belong to the class of feedback linearizable systems. On the other hand, as usually happens for nonlinear systems, the controllability property is hard to be established. In this paper we are pursuing a singular perturbation approach to the problem of controlling an elastic robot arm in order to use as much as possible the results already available in the rigid case.

A SINGULAR PERTURBATION MODEL

The generalized forces m(t) are the inputs to the system; thus there are actually N nonzero controls which we will call $(u_1(t), \dots, u_N(t))$ each of those corresponds to the motor of the i-th joint. Note that even though the system is highly nonlinear the controls enter linearly.

Let us explicetely split the vector q into two sets

$$q = (q_1^1, \dots, q_1^N, q_2^1, \dots, q_2^N)'$$

where

$q_1 = (q_1^1, \dots, q_1^N)'$ denotes the displacements of the driving bodies and

$q_2 = (q_2^1, \dots, q_2^N)'$ denotes the differences between the displacements of the driven bodies and of the drivinf ones. Eq.(2) can be rearranged accordingly as follows

$$\begin{aligned} \ddot{q}_1 &= f_1(q_1,q_2,\dot{q}_1,\dot{q}_2) + G_1(q_1,q_2)\, u \\ \ddot{q}_2 &= f_2(q_1,q_2,\dot{q}_1,\dot{q}_2) + G_2(q_1,q_2)\, u \end{aligned} \tag{5}$$

Let $k_1, \dots, k_N$ be the elastic constants of the N gear-boxes. One can presume that the orders of magnitude of the k_i's are comparable: hence a constant k can be factored so that $k_i = \hat{k}_i k$, $i = 1, \dots, N$; accordingly $k_1 q_2^1, \dots, k_N q_2^N$ are the elastic forces.

Inspecting the actual expressions of the entries of the matrices G_1 and G_2 and of the components of the vector fields f_1 and f_2 it turns out that eqs.(5) can be written more precisely as

$$\begin{aligned} q_1 &= f_1(kq_2,q_1+q_2,\dot{q}_1,\dot{q}_1+\dot{q}_2) + G_1(q_1,q_1+q_2)\, u \\ q_2 &= f_2(-kq_2,q_1+q_2,\dot{q}_1,\dot{q}_1+\dot{q}_2) + G_2(q_1,q_1+q_2)\, u \end{aligned} \tag{6}$$

Let us define $\xi_i=\hat{k}_i q_2^i$, i = 1, ... ,N and $z=k\xi$, i.e. $z = (k\xi_1, \dots ,k\xi_N)$ or $\xi = \varepsilon z$ where $\varepsilon = 1/k$. Clearly (q_1,z) is a perfectly good new state space. Substuting in eq.(6) one obtains

$$\begin{aligned} \ddot{q}_1 &= f_1(q_1+\varepsilon z,\ z,\dot{q}_1+\varepsilon\dot{z}) + G_1(q_1,q_1+\varepsilon z)\ u \\ \varepsilon\ddot{z} &= f_2(q_1+ z,-z,\dot{q}_1+\varepsilon\dot{z}) + G_2(q_1,q_1+\varepsilon z)\ u \end{aligned} \tag{7}$$

where f_2 and G_2 are suitably rescaled in order to absorbe the contants $\hat{k}_i$.

At this point our aim is to investigate the behaviour of the model (5) whenever k is large enough, i.e. with a small amount of elasicity (ε small enough).

As k goes to infinity, ε goes to zero and $\dot{q}_2$ (or $\dot{\xi}_i$, i=1, ... ,N) must go to zero as well, since there cannot be any finite velocity of the driven body with respect to the driving one in the rigid situation. As k goes to infinity, i.e. ε goes to zero, ξ must go to zero as well so that the elasic forces $z = k\xi$ do not become infinite, which is physically unfeasible. What is left to be established is which of the two following situations is compatible with the physical reality

$$\lim_{\varepsilon\to o} z = \lim_{\varepsilon\to o} \frac{\xi}{\varepsilon} \quad \text{is zero} \tag{8}$$

$$\lim_{\varepsilon\to o} z = \lim_{\varepsilon\to o} \frac{\xi}{\varepsilon} \quad \text{is finite} \tag{9}$$

In order for eqs.(7) to be a meaningful model they must return the equations which model the rigid case as $\varepsilon \to o$.

If eq.(8) holds eq.(7) gives

$$\begin{aligned} \ddot{q}_1 &= f_1(q_1,\dot{q}_1) + G_1(q_1)\ u \\ 0 &= f_2(q_1,\dot{q}_1) + G_2(q_1)\ u \end{aligned} \tag{10}$$

which does not model any physical situation. In fact the dynamic equation is completely controllable, being $G_1(q)$ nonsingular, and therefore there exists a feedback control law which induces a trajectory with constant speeds $\dot{q}_1^i=0$. If such a control law is substituted in the algebraic equation it follows that $q_1=\mathit{const}$ which is a contradiction; thus the case (8) is ruled out.

If eq.(9) holds eq.(7) gives

$$\begin{aligned} \ddot{q}_1 &= f_1(q_1,z,\dot{q}_1) + G_1(q_1)\ u \\ 0 &= f_2(q_1,-z,\dot{q}_1) + G_2(q_1)\ u \end{aligned} \tag{11}$$

The algebraic equation can be globally solved for z, since the elastic forces are additive according to Newton's laws. Substituting z in the first equation one actually obtains the rigid model equations

$$\ddot{q}_1 = f_R(q_1,\dot{q}_1) + B_R^{-1}(q_1)\ u \tag{12}$$

where

$$z = \psi(q_1,\dot{q}_1) + G_2(q_1)\ u$$

At this point the equations (7) describe a singular pertrbation problem which is formally well defined (6) since in the limit the model of the rigid case is exactly recovered.

If one investigates the rigid case in the extended state space as a special case of the more general elastic problem, the model is given by the constrained dynamical system

$$\begin{aligned} \ddot{q}_1 &= f_1(q_1,z,\dot{q}_1) + G_1(q_1)\ u \\ 0 &= f_2(q_1,-z,\dot{q}_1) + G_2(q_1)\ u \\ 0 &= \frac{d}{d\tau}(f_2(q_1,-z,\dot{q}_1) + G_2(q_1)\ u) \end{aligned} \tag{13}$$

The constraint, defined by the second and third algebraic, is a smooth submanifold whose dimensions are one half the extended state space dimensions.

As usual in singular perturbation approaches, one can define two time scales and rewrite eqs.(7) as

$$\begin{aligned} \frac{d^2 q_1}{dt^2} &= f_1(z,\ q_1+\varepsilon z,\dot{q}_1+\varepsilon\dot{z}) + G_1(q_1,\ q_1+\varepsilon z)\ u \\ \frac{d^2 z}{d\tau^2} &= f_2(-z,q_1+ z,\dot{q}_1+\varepsilon\dot{z}) + G_2(q_1,q_1+\varepsilon z)\ u \end{aligned} \tag{14}$$

where $\tau = t/\sqrt{\varepsilon}$.

A PROMISING CONTROL STRATEGY

At this point it is obviously interesting the behaviour of the fast dynamics around the constraint; to this purpose the linearization of the second equation around the manifold gives

$$\frac{d^2z}{d\tau^2} = \left(\frac{\partial f_2}{\partial z} + \frac{\partial(G_2u)}{\partial z}\right)\Bigg|_{\substack{z=\psi(q_1,\dot q_1)+G_2(q_1)u\\ \dot q_2=0}} (z - \psi(q_1,\dot q_1) + G_2(q_2)u) +$$

$$+ \frac{\partial f_2}{\partial z}\Bigg|_{\substack{z=\psi(q_1,\dot q_1)+G_2(q_1)u\\ \dot q_2=0}} \left(\dot z - \frac{d}{d\tau}(\psi(q_1,\dot q_1) + G_2(q_1)u)\right) \tag{15}$$

One can observe that the controls u affect the dynamics of (15) in two ways: directly through the Jacobian matrix

$$\left(\begin{array}{c|c} 0 & I \\ \hline \frac{\partial f}{\partial z} + \frac{\partial(G_2u)}{\partial z} & \frac{\partial f_2}{z} \end{array}\right) \tag{16}$$

and indirectly through the definition of the constraint.

It is apparent that there might be control laws such that the fast dynamics is unstable in a certain region of the state space.

For instance suppose we know how to design a control law $\bar u$ such that the rigid model produces a preassigned trajectory. In the extended state space, if the real system is completely rigid, the trajectory belongs to the constraint

$$z = \psi(q_1,\dot q_1) + G_2(q_1)\,\bar u$$

$$\dot z = \frac{d}{d\tau}(\Psi(q_1,\dot q_1) + G_2(q_1)\,\bar u$$

If, on the other hand, the real system is not rigid, the trajectory produced by the elastic model in correspondence to the control law $\bar u$

asymptotically tracks the trajectory produced by the rigid model if the matrix (16) is stable around the "rigid" trajectory.

We already showed that the rigid model is controllable and that a state feedback control can be explicitely obtained (eq.(3)). If the system is weakly elastic, simulation experiences prove that such a control law might not work, depending on the stability properties of the matrix (16) (see (5) pag.288).

However, being the system weakly elastic, it can be described by eqs.(14). One can observe that the dipendence of the feedback control law on the "elastic" state variables z and $\dot{z}$ greatly affects the stability properties of matrix (16). In the case one succeeds using such a tool in making the matrix (16) stable in the region of interest, the slow equation would only be perturbed by constant disturbances, which can be more easily compensated.

Thus one might think of adding a fast state feedback control law to the slow state feedback control law previously determined according to such a control scheme.

In conclusion we propose the following control law for robots which be significantly represented by eqs.(14)

$$u = u_s(q_1,\dot{q}_1,v) + u_f(q_1,z,\dot{z}) = B_R(q_1)v(t) - B_R(q_1)f_R(q_1,\dot{q}_1) + + B_R(q_1)w(z,\dot{z})$$

where u_s is the "slow" part and u_f is the "fast" part of the overall control u; v represents the new inputs and the subscript R identifies the rigid model. As stated before $w(z,\dot{z})$ must be chosen in order to guarantee the stability of the matrix (16) in the region of the interest.

One finally obtains the following controlled system

$$\frac{d^2q_1}{dt^2} = f_1(z,q_1+\varepsilon z,\dot{q}_1+\varepsilon\dot{z}) - G_1(q_1,q_1+\varepsilon z)B_R(q_1)f_R(q_1,\dot{q}_1) + + G_1(q_1,q_1+\varepsilon z)B_R(q_1)v(t) + G_1(q_1,q_1+\varepsilon z)B_R(q_1)w(z,\dot{z})$$

$$\frac{d^2z}{d\tau^2} = f_2(-z,q_1+\varepsilon z,\dot{q}_1+\varepsilon\dot{z}) + G_2(q_1,q_1+\varepsilon z)u_s(q_1,\dot{q}_1,v) + + G_2(q_1,q_1+\varepsilon z)\ B_R(q_1)w(z,\dot{z})$$

where $w(z,\dot{z})$ and $v(t)$ are to be designed.

ACKOWLEDGEMENTS

The authors wish to thank professor F. Nicolò for some helpful suggestions and discussions.

In addition the authors wish to acknowledge the support of this work by CNR (Progetto Finalizzato Informatica) and by MPI (fondi 40%).

REFERENCES

1. G. Cesareo, F. Nicolò, S. Nicosia - Dynamical Models of Industrial Robots - *First IASTED Int. Symp. on Applied Modeling and Simulation* - Lyon, sept. 1981.
2. S. Nicosia, F. Nicolò, D. Lentini - Dynamical Control of Industrial Robots with Elastic and Dissipative Joints - *8th IFAC Triennial World Congress* - Kyoto, aug. 1981.
3. B. Jakubczyk, W. Respondek - On Linearization of Control Systems - *Bull. de l'Academie Polonaise des Sciences* - vol. XXVIII, n°9-10, 1980.
4. L.R. Hunt, R. Su, G. Meyer - Design for Multiinput Nonlinear Systems - *Differential Geometric Control Theory (R. Brockett et alii ed.)* - Birkhäuser, 1983.
5. W. Hahn - *Stability of Motion* - Springer Verlag, 1967.
6. G. Peponides, P.V. Kokotovic - Weak Connections, Time Scales, and Aggregation of Nonlinear Systems - *IEEE Transactions on Circuits and Systems* - 1982

DECENTRALIZED SUCCESSIVE APPROXIMATION METHODS FOR OPTIMAL CONTROL

J.L. CALVET
L.A.A.S. du C.N.R.S., Toulouse, France.

Abstract : Based on partitioning and relaxation strategies, iterative methods are proposed for a distributed computation of large scale differential systems. Extensions of original splitting methods in R^n [1] [2] are investigated to solve Initial-Value-Problems (IVP), and are proved to increase convergence efficiency on account of the Volterra operators properties. In the field of optimal control it is shown how, decentralized schemes of optimization apply themselves more effectively to problems imbed in a more appropriate class of IVP, than to the original Two-Point-Boundary-Value-Problem (TPBVP) of stationarity conditions, which is usual in hierarchical control [3][4].

DECENTRALIZED INTEGRATION OF LINEAR SYSTEMS

A large scale differential linear system is considered. Based on a partition of the system matrix, several block-decomposition methods are defined as an extension of splitting methods [1] into the framework of decomposition in function space. The basic schemes of parallel (sequential) decomposition, respectively associated with Jacobi (Gauss Seidel) algorithms of coordination, can be easily defined.

Convergence is analyzed by referring to convergence theorems of bounded linear operators in metric space [5] and to P-contraction mapping theorem [2]. Application of the generalized P-contraction mapping theorem [6] enables one to theoretically guaranty the convergence of decentralized successive approximation from an arbitrary initial guess and for any choice of partition.

PARTITIONING FOR OPTIMIZATION OF CONTROL SYSTEMS

Concerning the quadratic optimization of linear systems, two kinds of partition are considered.

Partition 1 takes profit from the powerful theory of numerical analysis and broads considerably the scope of decentralized methods for optimal control.

Partition 2 does correspond to the more recent hierarchical control theory. Such partitions have been tentatively compared. To summarize, let us say that Partition 1 has the advantage to involve IVP at the lower level. It is well adapted to the case where state and costate equations are weakly coupled. In the case of Partition 2, the subproblems are defined in terms of TPBVP and correspond to local optimization subproblems. This case is more devoted to weakly coupled physical systems.

The salient fact of this paper is that decomposition must be set not only on the overall problem partitioning, the size of subsystem or the type of processing (parallel, sequential, asynchronous...), but also on the type of subproblems. Thus, we successively consider decentralized approximation to solve :

- the reduced canonical system of stationarity conditions (TPBVP)
- the linear fundamental system (IVP)
- the Riccati differential equation.

When solving the linear fundamental system, it is emphasized that, although one deals with an expanded differential system, one can profit of getting an IVP to solve it in a distributed fashion. Moreover this system, which appears as a set of independent systems, permits composite decomposition.

In front of the intrinsic instability of Hamiltonian systems, decomposition is finally extended to the non linear Riccati differential equation.

Experimentation on a power system with stiff equations has shown satisfactory results by using decomposition methods, especially when solving the Riccati equation.

References

[1] R.S. VARGA, Matrix iterative analysis, Prentice Hall, 1962.

[2] J.M. ORTEGA and W.C. RHEINBOLDT, Iterative solutions of non linear equations in several variables, Academic Press, 1970.

[3] M.D. MESAROVIC, D. MACKO and Y. TAKAHARA, Theory of hierarchical multilevel systems, Academic Press, 1970.

[4] G. COHEN, Optimization by decomposition and coordination : a unified approach. IEEE Trans. on T.A.C., vol. AC-23, n° 2, 222-232, 1978.

[5] A.J. LAUB and F.N. BAILLEY, An iterative coordination approach to decentralized decision problems, IEEE T.A.C., vol. AC-23, n° 6, 1031-1036, 1978.

[6] D.G. LUENBERGER, Optimization by vector space methods, Wiley, 1969.

A DIAGNOSIS ALGORITHM FOR THE BGM SYSTEM LEVEL FAULT MODEL

G. G. L. MEYER
Electrical Engineering and Computer Science Department
The Johns Hopkins University
Baltimore, Maryland 21218

ABSTRACT

A τ-diagnosable system is a system in which all faults may be identified from the test results, provided that the number of faults does not exceed τ. In this paper we present an algorithm that may be used for the diagnosis of the system level BGM fault model proposed by Barsi, Grandoni and Maestrini, whenever the system is τ-diagnosable and the number of faults is at most τ.

THE BGM SYSTEM LEVEL FAULT MODEL

Consider a system S of n units $u_1, u_2, \ldots, u_n$ and a test digraph TD, where $TD = \{ (u_i, u_j) \mid u_i \text{ tests } u_j \}$. It is assumed that no unit tests itself, that each unit is either faulty or nonfaulty, and that the state of each unit is constant during the application of the testing procedures. If (u_i, u_j) is in TD, then u_i tests u_j, and the test outcome a_{ij} is assumed to be either "0" (u_j passes the test) or "1" (u_j fails the test). The set of test outcomes $\{ a_{ij} \mid (u_i, u_j) \in TD \}$ is the syndrome of the system. In the BGM model proposed by Barsi, Grandoni and Maestrini [1], the following relationships between faults and test outcomes are assumed:

(i) if (u_i, u_j) is in TD and u_i and u_j are nonfaulty, then $a_{ij} = 0$;

(ii) if (u_i, u_j) is in TD, u_i is nonfaulty and u_j is faulty, then $a_{ij} = 1$;

(iii) if (u_i, u_j) is in TD and both u_i and u_j are faulty, then $a_{ij} = 1$;

(iv) if (u_i, u_j) is in TD, u_i is faulty and u_j is nonfaulty, then a_{ij} may take either the value 0 or 1.

Thus, if a unit u_i is tested by a unit u_j and $a_{ji} = 0$, the unit u_i is nonfaulty.

Given a set of faulty units F_S, the computation of the corresponding syndromes is not difficult, but to compute the sets of faulty units that are consistent with a given syndrome is not as easy. In this paper, we address the latter problem -- namely, syndrome decoding -- and we restrict ourselves to τ-diagnosability in the sense of Preparata, Metze and Chien [6].

Definition 1: A system S is τ-diagnosable if all faulty units within the system can be identified without replacement, provided that the number of faulty units does not exceed τ.

In the remainder of this work, $\|A\|$ will be used to denote the number of elements in the set A.

FAULT IDENTIFICATION ALGORITHM

Our approach to system diagnosis consists in defining subsets V, H_1, H_2 and H_3 that depend on the syndrome and two subsets W and X that depend only on the test digraph, and then to relate those subsets to the set F_S of faulty units in S.

The set V contains all the units in S that are tested by at least one other unit in S and found to be nonfaulty by that unit, i.e.,

$$V = \{ u_i \in S \mid u_j \text{ in } S \text{ exists so that } (u_j,u_i) \in TD \text{ and } a_{ji} = 0 \}. \qquad (1)$$

Thus, if S is a BGM model, the unit u_i is nonfaulty whenever u_i is in V.

The set H_1 contains all the units in S that are tested by at least one unit u_j in V and found faulty, and all the units in S that test at least one unit u_j in V, and find it faulty, i.e.,

$$\begin{aligned} H_1 = \{u_i \in S \mid\, & u_j \text{ in } V \text{ exists so that } (u_j,u_i) \in TD \text{ and } a_{ji} = 1 \} \\ \cup \{u_i \in S \mid\, & u_j \text{ in } V \text{ exists so that } (u_i,u_j) \in TD \text{ and } a_{ij} = 1 \}. \qquad (2) \end{aligned}$$

One should note that if S is a BGM model, then the sets V and H_1 are disjoint, and u_i is faulty whenever u_i is in H_1.

The index set H_2 depends on the cardinality of the sets $L(u_i)$, where, for every unit u_i in $S - (V \cup H_1)$, the sets $L(u_i)$ are defined by

$$\begin{aligned} L(u_i) = \{ u_j \in S - (V \cup H_1) \mid (u_i,u_j) \in TD \text{ and } a_{ij} = 1 \} \\ \cup \{ u_j \in S - (V \cup H_1) \mid (u_j,u_i) \in TD \text{ and } a_{ji} = 1 \}. \end{aligned}$$

Given u_i, it is possible that u_j exists so that (u_i,u_j) and (u_j,u_i) are both in TD, and $a_{ij} = a_{ji} = 1$. Obviously, in such a case u_j appears in $L(u_i)$ only once. The set $L(u_i)$ contains all the units adjacent to the unit u_i that must be faulty if the unit u_i is actually nonfaulty. Given a scalar τ, the set H_2 consists of all the units in S, but not in $V \cup H_1$, such that the cardinality of $L(u_i)$ is strictly greater than τ, i.e.,

$$H_2 = \{ u_i \in S - (V \cup H_1) \mid \|L(u_i)\| \geqslant \tau + 1 \}. \tag{3}$$

It is clear that if S is a BGM model and if at most τ units in S are faulty, then u_i is faulty whenever it is in H_2.

The set H_3 contains the remaining units in S, i.e.,

$$H_3 = S - (V \cup H_1 \cup H_2).$$

The definition of the sets H_1, H_2 and H_3 immediately implies the following lemma.

Lemma 1: If (i) S is a BGM model, and (ii) $\|F_S\| \leqslant \tau$, then

$$H_1 \cup H_2 \subseteq F_S \subseteq H_1 \cup H_2 \cup H_3.$$

The two subsets W and X of S that are defined now depend only on the test digraph TD and do not depend on the syndromes produced by faulty sets of units. Note that the subset W is not used in the fault identification algorithm, and is defined only to facilitate the analysis of the algorithm.

The set W contains all the units u_i in S such that: (i) the unit u_i is tested by exactly τ other units, and (ii) a unit u_j in S exists such that u_j is tested by exactly τ other units in S, and u_i and u_j test each other.

The set X be the set of all units u_i in S such that: (i) u_i is tested by exactly τ other units; (ii) a unit u_j exists such that u_j is tested by exactly τ units in S and u_i and u_j test each other; and (iii) a unit u_k in S exists such that the unit u_k tests u_j but not u_i, and u_k is tested by at least one unit that does not test u_i.

We are now ready to present our fault identification algorithm.

Algorithm 1:

Step 0: Compute the set V as in Equation (1).

Step 1: If $\|S - V\| \leqslant \tau$, let $F_A = S - V$ and stop; otherwise, go to Step 2.

Step 2: Compute the sets H_1 and H_2 as in Equations (2) and (3).

Step 3: If $\|H_1 \cup H_2\| = \tau$, let $F_A = H_1 \cup H_2$ and stop; otherwise, go to Step 4.

Step 4: Let $F_A = H_1 \cup H_2 \cup (H_3 \cap X)$ and stop.

GORITHM ANALYSIS

We start the analysis of Algorithm 1 by presenting its properties when the following assumption is satisfied.

Hypothesis 1: Every unit in S is tested by at least τ other units in S.

We now show that when Hypothesis 1 is satisfied, the set F_A generated by Algorithm 1 contains only faulty units.

Lemma 2: If (i) S is a BGM model, (ii) Hypothesis 1 is satisfied, and (iii) $\|F_S\| \leqslant \tau$, then $F_A \subseteq F_S$.

Proof: (i) Assume that $\|H_1 \cup H_2 \cup H_3\| \leqslant \tau$. In that case, Algorithm 1 stops in Step 1 and $F_A = S - V = H_1 \cup H_2 \cup H_3$. Two cases are possible: either $\|F_S\| < \tau$ or $\|F_S\| = \tau$.

(i.a) Assume that $\|F_S\| < \tau$. Let u_i be a nonfaulty unit. By assumption, every unit is tested by at least τ other units and the fact that $\|F_S\| < \tau$ implies that u_i is tested by at least one nonfaulty unit, say u_j. Thus, if u_i is nonfaulty, a unit u_j exists so that (u_j, u_i) is in TD, $a_{ji} = 0$, and it follows that V contains the indices of all the nonfaulty units in S. Now let u_i be a faulty unit. Hypothesis 1 and the fact that $\|F_S\| < \tau$ imply that u_i is tested by at least one unit u_j in V and $a_{ji} = 1$. It follows that $H_1 = F_S$, V and H_1 form a partition for S, $S - V = H_1$, and thus $F_A = F_S$.

(i.b) Assume now that $\|F_S\| = \tau$. In that case, Lemma 1 implies immediately that $F_S = H_1 \cup H_2 \cup H_3 = S - V = F_A$.

(ii) Assume now that $\|H_1 \cup H_2 \cup H_3\| > \tau$. By assumption, $\|F_S\| \leqslant \tau$, and thus, using Lemma 1, we may conclude that $\|H_1 \cup H_2\| \leqslant \tau$. Thus, once again, two cases are possible: either $\|H_1 \cup H_2\| = \tau$ or $\|H_1 \cup H_2\| < \tau$.

(ii.a) Assume that $\|H_1 \cup H_2\| = \tau$. In that case, Algorithm 1 stops in Step 3 and $F_A = H_1 \cup H_2$. Lemma 1 implies immediately that $\|F_S\|$ must be equal to τ and that $F_A = F_S$.

(ii.b) Assume now that $\|H_1 \cup H_2\| < \tau$. In that case, Algorithm 1 stops in Step 4 and $F_A = H_1 \cup H_2 \cup (H_3 \cap X)$. Let u_i be in $H_3 \cap X$ and let u_j and u_k be units that satisfy part (ii) and (iii) in the definition of the set X. Suppose that u_i and u_k are both nonfaulty. All the units that test u_i and those that test u_k are then faulty. The unit u_k is tested by at least one unit that does not test u_i and therefore, the assumption that both u_i and u_k are nonfaulty implies that at least $\tau + 1$ units are faulty. This is impossible, and thus, if u_i is nonfaulty, u_k must be faulty. The unit u_k does not test the unit u_i, and the fact that u_i is in H_3 implies that u_i does not test u_k. Thus, if u_i is assumed to be nonfaulty, we must conclude that at least $\tau + 1$ units are faulty. Once again, this is impossible; Therefore, the unit u_i must be faulty -- i.e., u_i must be in F_S. We already know from Lemma 1 that H_1 and H_2 are subsets of F_S, and thus, we may conclude that $H_1 \cup H_2 \cup (H_3 \cap X)$ is in F_S, i.e., that F_A is a subset of F_S.

In their 1976 paper, Barsi, Grandoni and Maestrini [1] proposed a condition on the test digraph TD that insures τ-fault diagnosability. Using our notation, we will now repeat that assumption and show that it may be used to insure that F_S is found by Algorithm 1

Hypothesis 2: If the units u_i and u_j are in W and if u_i and u_j test each other, then u_i or u_j or both are in X.

Lemma 3: If (i) S is a BGM model, (ii) Hypotheses 1 and 2 are satisfied, (iii) $\|F_S\| \leqslant \tau$, (iv) $\|H_1 \cup H_2 \cup H_3\| > \tau$ and (v) $\|H_1 \cup H_2\| < \tau$, then $H_3 \subseteq W$, $\|F_S\| = \tau$ and $H_1 \cup H_2 \cup (H_3 \cap X) = F_S$.

Proof: (i) Let u_i be a unit in H_3. Every unit in S that tests u_i find u_i faulty (otherwise u_i would be in V). No unit in V tests u_i (otherwise u_i would be in H_1).

The unit u_i cannot be tested by more than τ other units (otherwise u_i would be in H_2), and thus, Hypothesis 1 implies that u_i is tested by exactly τ other units. We may conclude that every unit in H_3 is tested by exactly τ other units that must be in $H_1 \cup H_2 \cup H_3$.

We have assumed that $\|H_1 \cup H_2\| < \tau$, and therefore, if u_i is in H_3, u_i must be tested by at least one other unit u_j also in H_3. If u_i does not test u_j, then u_j is tested and found faulty by a set of τ units that does not include u_i, and u_j tests and finds u_i faulty. It is clear that $L(u_j) > \tau$, and thus u_j must be in H_2. This contradicts the fact that u_j is in H_3 and therefore we may conclude that u_i tests u_j. Every unit in H_3 is in W and thus $H_3 \subseteq W$.

(ii) Suppose that $\|F_S\| < \tau$. Part (i.a) of the proof of Lemma 2 shows that $H_1 = F_S$ and V and H_1 form a partion for S. Thus, H_2 and H_3 are empty and $\|H_1 \cup H_2 \cup H_3\| < \tau$. This contradicts the fact that $\|H_1 \cup H_2 \cup H_3\| > \tau$, and we may conclude that $\|F_S\| = \tau$.

(iii) Suppose that a unit u_i exists so that u_i is in H_3 but is not in X. The unit u_i is tested by exactly τ other units that must be in $H_1 \cup H_2 \cup H_3$. It follows that u_i is tested by at least $\tau - \|H_1 \cup H_2\|$ units in H_3. Hypothesis 2 then implies that all those units in H_3 (and thus in W) that test u_i must be in X. We may then conclude that $\|H_3 \cap X\| = \tau - \|H_1 \cup H_2\|$. The fact that the sets H_1, H_2 and H_3 are disjoint then implies that $\|H_1 \cup H_2 \cup (H_3 \cap X)\| = \tau$. We have proved that $\|F_S\| = \tau$, from Lemma 2, we know that $\|H_1 \cup H_2 \cup (H_3 \cap X)\| \subseteq F_S$, and it follows that $H_1 \cup H_2 \cup (H_3 \cap X) = F_S$.

(iv) Suppose that all units in H_3 are also in X. In that case, $H_3 \cap X = H_3$ and Lemma 1 implies immediately that $H_1 \cup H_2 \cup (H_3 \cap X) = F_S$.

Using Lemmas 2 and 3, we may then obtain the following result.

Theorem 1: If (i) S is a BGM model, (ii) Hypotheses 1 and 2 are satisfied, and $\|F_S\| \leqslant \tau$, then the set F_A generated by Algorithm 1 is equal to F_S.

It is known that if a BGM model is τ-diagnosable, then Hypotheses 1 and 2 are satisfied [1]. Hence, we obtain the main result of the paper.

Theorem 2: Let S be a τ-diagnosable BGM fault model and let F_S be the set of faulty units in S. If $\|F_S\| \leqslant \tau$, then the set F_A generated by Algorithm 1 is equal to F_S.

Reference [5] contains a comprehensive bibliography concerning system level fault models and some additional results concerning Algorithm 1. For example, it is shown that if S is a BGM model, Hypothesis 1 is satisfied and no two units test each other [2, Theorem 1], then Hypothesis 2 is automatically satisfied and Algorithm 1 always stop in either Step 1 or 3 whenever $\|F_S\| \leqslant \tau$. Note that Holt has obtained diagnosability results and some diagnosis algorithms for a system level fault model that is related to the BGM model [3], [4].

ACKNOWLEDGEMENT

This work was supported by the Office of Naval Research under Contract N00014-80-C-0772.

REFERENCES

[1] Barsi, F., Grandoni, F., and Maestrini, P., A Theory of Diagnosability of Digital Systems, *IEEE Trans. Computers,* Vol. C-25 , June 1976, pp. 585-593.

[2] Hakimi, S.L., and Amin, A.T., Characterization of Connection Assignment of Diagnosable Systems, *IEEE Trans. Computers,* Vol. C-23 , January 1974, pp. 86-88.

[3] Holt, C.S. and Smith, J.E., Diagnosis of Systems with Asymmetric Invalidation, Department of Electrical and Computer Engineering, Technical Report ECE-79-18, University of Wisconsin, Madison, 1979.

[4] Holt, C.S., Diagnosis and Self-Diagnosis of Digital Systems, Ph. D. Dissertation, University of Wisconsin, Madison, 1981.

[5] Meyer, G.G.L., One-Step Diagnosis Algorithms for the BGM System Level Fault Model, Department of Electrical Engineering and Computer Science, Report JHU/EECS-82/14, The Johns Hopkins University, Baltimore, 1982.

[6] Preparata, F.P., Metze, G., and Chien, R.T., On the Connection Assignment Problem of Diagnosable Systems, *IEEE Trans. Electronic Computers,* Vol. EC-16 , December 1967, pp.848-854.

ADAPTIVE FAULT IDENTIFICATION IN TWO NEW DIAGNOSTIC MODELS*

S. E. KREUTZER and S. L. HAKIMI
Department of Electrical Engineering and Computer Science
Northwestern University, Evanston, Illinois 60201

ABSTRACT

In the area of system-level diagnosis, in particular t-diagnosable systems, the PMC and BGM models have received considerable attention. In this paper, shortcomings of these models are discussed, two viable alternative models are introduced, the test results needed for diagnosis are characterized for each model, and efficient diagnosis algorithms are presented.

INTRODUCTION

The increasing size and complexity of computing systems along with the need for high availability and reliability has necessitated the development of automatic fault diagnosis for system maintenance. As a consequence, a fundamental theory of system-level diagnosis has evolved. Basic to this theory is the diagnostic model which represents the units (machines) which comprise a computing system, the interconnections between the units, and the implications of the diagnostic test results. The PMC model [1], proposed by Preparata, Metze, and Chien, has received the most attention. The BGM model [2], which is similar, has also been widely studied. In both models, a system is represented as a digraph, G(V,E), in which the vertices correspond to the units which comprise the computing system and the edges represent the tests which can be performed.

More precisely, let $V = \{u_1,u_2,\ldots,u_n\}$ be the set of n units and $E = \{e_1,e_2,\ldots,e_q\}$ be the possible tests. A test $e_i = (u_j,u_k)$ corresponds to the test that unit u_j can perform on unit u_k. Furthermore, the result of test e_i is represented as the edge-weight $w(e_i) = w(u_j,u_k)$. The edge-weight may be 0 or 1 corresponding to tester u_j evaluating u_k as fault-free or faulty, respectively. Furthermore, a system is defined to be t-diagnosable if and only if based on the test results, it can unambiguously be decided whether each of the n units in the system is faulty or fault-free provided that the number of faulty units does not exceed some given number t.

Regarding the implications of the test results, in the PMC model it is assummed that a fault-free tester's evaluation is always correct, whereas a faulty tester's evaluation may or may not be correct, regardless of the condition of the tested unit. In the BGM model, the test results are based on the premise that a diagnostic evaluation (or test) involves a large

---

* This work was supported by the National Science Foundation under grant ECS-8201387.

number of tests (stimuli) and the observation of the corresponding responses. If the tested unit provides the responses expected by the tester, the tester evaluates the unit as fault-free. Otherwise, the tested unit is evaluated as faulty. The BGM model is based on the assumption that if a tested unit is faulty even if the tester is faulty there is likely to be a mismatch between the actual and expected responses.

For each of the models, two widely studied problems are the characterization of the topologies of t-diagnosable systems and the discovery of optimal algorithms in terms of the number of tests needed for system diagnosis. In this paper, shortcomings of the PMC and BGM models are discussed and the viabilities of two new models which overcome the shortcomings are established. Furthermore, characterization theorems and optimal or efficient adaptive diagnosis algorithms are provided for each new model.

THE FIRST MODEL

In regard to the test results, the assumptions made in the PMC model may be too strict. In the PMC model, it is assummed that a faulty tester's evaluation may or may not be correct. But assumming a diagnostic evaluation involves a large number of tests, the responses a faulty tester expects will not agree with the responses provided by a fault-free unit. However, when a faulty tester evaluates a faulty unit the two units may agree on test results when both units are faulty in exactly the same manner. The first model we propose is based on these assumptions. In this model, a test result of zero implies that either both units are faulty or both units are fault-free, while a test result of one implies that either or both units are faulty.

Before presenting the necessary and sufficient conditions for t-diagnosability, some definitions are given. Let $|X|$ denote the cardinality of set X. Let $G(V,E)$ be an undirected graph. Let $\Gamma(u) = \{v \varepsilon V \mid (u,v) \varepsilon E\}$ and for set $U \subset V$, let $\Gamma(U) = \bigcup\{\Gamma(v) \mid v \varepsilon U\} - U$. Furthermore, $X \subset V$ is a consistent fault set if and only if it is consistent with the test results that X contains only faulty units and that $V-X$ contains only fault-free units. A set $X \subset V$ is a consistent fault set in this model if and only if (1) for all $u \varepsilon X$ and $v \varepsilon V-X$, $w(u,v)$ and $w(v,u) = 1$ and (2) for all $\{u,v\} \subseteq V-X$, $w(u,v) = 0$. Note that a system is t-diagnosable if and only if given the test results there exists only one consistent fault set of cardinality at most t. Let G be a digraph. $G'(V,E)$, formed from G, is called the underlying undirected graph if the directions on the edges in G are ignored and the parallel edges in G are represented by a single edge in G'.

Theorem 1 Let G be the digraph representing a system S of n units under the first model. Let $G'(V,E)$ be the underlying undirected graph. S is t-diagnosable if and only if (1) $n \geqslant 2t + 1$, (2) for all u in V, $|\Gamma(u)| \geqslant t$, and (3) for each integer p with $0 \leqslant p \leqslant t$ and each $|X| = n-2t+p$, $|\Gamma(X)| \geqslant p$.

Proof: The proof of the necessity of conditions (1) and (2) will not be shown since it can easily be adapted from the proof of the characterization theorem for the PMC model [1]. To prove the necessity of condition (3), we will show that if $|X| = n-2t+p$ with $0 \leqslant p \leqslant t$ and $|\Gamma(X)| \leqslant p$, then for some test results there exist two distinct consistent fault sets. Let $q = |\Gamma(X)|$ and $Z = V - (X \cup \Gamma(X))$, thus $|Z| = n - (n-2t+p+q) = 2t-p-q \geqslant 2$. Divide Z into two non-empty sets (Z_1 and Z_2) whose cardinalities differ by at most one and assume $|Z_1| \geqslant |Z_2|$. Let X contain only fault-free units and $\Gamma(X)$ contain only faulty units. Since $\Gamma(X) \cap (Z_1 \cup Z_2) = \emptyset$, assume that all tests of units in $Z_1 \cup Z_2$ have result one. Thus, we have two distinct consistent fault sets: $F_1 = \Gamma(X) \cup Z_1$ and $F_2 = \Gamma(X) \cup Z_2$. It remains to be shown that the cardinalities of F_1 and F_2 are at most t. $|F_1| = |\Gamma(X)| + \lceil Z/2 \rceil = q + \lceil (2t-p-q)/2 \rceil \leqslant q + t - \lceil (p+q)/2 \rceil$. Since $q \leqslant p$, $|F_1| \leqslant q + t - \lceil (q+q)/2 \rceil \leqslant t$. Since $|Z_2| \leqslant |Z_1|$, the necessity of (3) is proven.

To prove the sufficiency, assume that conditions (1)-(3) hold for system S and that S is not t-diagnosable. Then there exist two distinct consistent fault sets F_1 and F_2 of cardinality $\leqslant t$. Let $F_{12} = F_1 \cap F_2$, $Z_1 = F_1 - F_{12}$, $Z_2 = F_2 - F_{12}$, $X = V-(F_1 \cup F_2)$. It is first shown that $n-2t \leqslant |X| \leqslant n-t$. Since $|F_1| = |Z_1| + |F_{12}| \leqslant t$ and $|F_2| = |Z_2| + |F_{12}| \leqslant t$, then

$$|X| = n - (|Z_1| + |Z_2| + |F_{12}| \geqslant n - 2t + |F_{12}| \geqslant n - 2t \quad (4).$$

Next, to show that $|X| \leqslant n - t$, assume otherwise. Then $|X| \geqslant n-t$, which implies that $|V-X| = |F_1 \cup F_2| \leqslant t$. Let $v \in Z_1 \cup Z_2$. $\Gamma(v) \cap X$ is empty because any test result between v and a unit in X would contradict the definitions of the consistents fault sets. Thus, $\Gamma(v) \subseteq F_1 \cup F_2$. But $|F_1 \cup F_2| \leqslant t$ and consequently $|\Gamma(v)| < t$. This contradicts condition (2) of the characterization theorem. Hence, $|X| \leqslant n-t$. Thus there exists an integer p, with $0 \leqslant p \leqslant t$ such that $|X| = n-2t+p$. Let $q = |F_{12}| \geqslant |\Gamma(X)| \geqslant p$. Then by (4), $|X| \geqslant n-2t+|F_{12}| = n-2t+q$. But since $q \geqslant p$ this is a contradiction.

Condition (3) is difficult to verify. Hence, we shall provide simpler sufficient conditions for t-diagnosability. But before doing so, recall that the vertex connectivity, k(G), of an undirected graph G(V,E) is the minimum number of vertices whose removal from G yields a graph which is not connected.

Corollary: Let G be the digraph representing system S of n units under the first model. Let G'(V,E) be the underlying undirected graph. If (1) $n \geqslant 2t + 1$ and (2) $k(G') \geqslant t$, then S is t-diagnosable.

Proof: We need only to show that condition (3) of Theorem 1 is satisfied. Assume otherwise: For $|X| = n-2t+p$, $|\Gamma(X)| \leqslant p \leqslant t$. Consequently, removing $p \leqslant t-1$ vertices disconnects G'. This is a contradiction and the corollary is proven.

Some very interesting conclusions can be drawn from the characterization theorem. For instance, it is not difficult to see that any system which is t-diagnosable under the PMC model is also t-diagnosable under the first model. Furthermore, indegree is not important in this model. In fact, the direction of the edges is irrelevant. This has important implications for one-step diagnosability. For example, let G(V,E) be a graph without loops and parallel edges, $|V| \geqslant 2t + 1$ and $|\Gamma(u)| \geqslant t$ for all u in V. For any digraph G' obtained from G by orienting its edges, G' is a t-diagnosable system in our model, while G' is at best t/2-diagnosable in the PMC model.

AN ADAPTIVE ALGORITHM FOR SYSTEM DIAGNOSIS

In this section we shall present an adaptive algorithm which identifies the stautus of every unit in the system. In adaptive diagnosis, originally introduced into the area of t-diagnosable systems by Nakajima [3], tests are chosen one at a time based on previous test results. It is unlike one-step diagnosis in which all of the tests are chosen before beginning diagnosis. Steps 0-3 were adapted from Hakimi and Schmeichel's algorithm in [4].

Algorithm 1.

0. Construct a list L_1 consisting of one-sequences which initially corresponds to the set of vertices, V. Introduce lists L_2, L_3 and L_4, which are initially empty. Each k-sequence, $t_i = (u_1,u_2,\ldots,u_k)$ represents a directed path from u_1 to u_k. Let $|L_1|$ denote the number of sequences in L_1, and $|t_i|$ denote the length of sequence t_i. In addition let f, initially zero, be the number of faulty units which have been isolated to a set of 2f units in L_3. Let k denote the length of the first sequence in L_1.

2. Recursively take the first two members of L_1 ($t_i = (u_1,u_2,\ldots,u_k)$ and $t_j = (v_1,v_2,\ldots,v_k)$), delete them from L_1, seek $w(u_k,v_1)$, and create the

2k-sequence $t_q = (u_1,\ldots,u_k,v_1,\ldots,v_k)$ until either $|L_1| \leq 1$ or $|t_i| > |t_j|$. If $w(u,v) = 0$, place t_q at the top of L_2. If $w(u,v) = 1$, set $f \leftarrow f + k$ and place t_q in L_3. If $k + f > t$, then go to 4.

3. Set $L_1 \leftarrow (L_2$ followed by $L_1)$ and $L_2 \leftarrow \emptyset$. If $k+f \leq t$, then go to step 2.

4. All sequences t_i in $L_1 \cup L_2$ for which $|t_i| > t-f$ contain only fault-free units. Arbitrarily choose a tester, r, from these units. For all sequences $t_i = (u_1,\ldots,u_j)$ in L_1 for which $|t_i| \leq t-f$, seek $w(r,u_1)$. If the result is zero (one), then t_i contains only fault-free (faulty) units. If the result was one, let $f \leftarrow f+j$.

5. At any point, if $f = t-1$, go to 7. For all sequences $t_i = (u_1,\ldots,u_j,u_{j+1},\ldots,u_{2j})$ in L_3 remove t_i and seek $w(r,u_1)$. If $w(r,u_1) = 0$, then units $u_1,\ldots,u_j$ are fault-free and units $u_{j+1},\ldots,u_{2j}$ are faulty. If $w(r,u_1) = 1$, then units $u_1,\ldots,u_j$ are faulty. If $f+j > t$, all units $u_{j+1},\ldots,u_{2j}$ are fault-free. Otherwise, $(f+j \leq t)$ seek $w(r,u_{j+1})$. If the test result is zero (one), then $u_{j+1},\ldots,u_{2j}$ are fault-free (faulty and let $f \leftarrow f + j$).

6. Stop.

7. For all $t_i = (u_1,\ldots,u_j,u_{j+1},\ldots,u_{2j})$, remove t_i from L_3 and seek $w(r,u_j)$. If the result is zero all $u_1,\ldots,u_j$ are fault-free and $u_{j+1},\ldots,u_{2j}$ are faulty. If the result is one, all $u_1,\ldots,u_j$ are faulty, and if $j > 1$, $u_{j+1},\ldots,u_{2j}$ are fault-free, otherwise place u_{j+1} in L_4.

8. Pick u and v in L_4, remove u and v, and seek $w(u,v)$, until $w(u,v) = 1$. If the result is one, then go to step 9. Otherwise, u and v are fault-free. If there remains a single unit x, seek $w(r,x)$. Stop.

9. Seek $w(r,u)$ and stop. If $w(r,u) = 0$ (1) then, u (v) is fault-free and v (u) is faulty and all units in L_4 are fault-free.

Before proving that the algorithm correctly determines the status of every unit in the system, some preliminary observations are made.

(1) No unit is a member of more than one list.

(2) Lists L_1 and L_2 contain sequences which correspond to paths of zero-weight edges and because of the assumptions on which the test results are based, all units in the sequence are faulty or all are fault-free.

(3) Consider the sequence $t_q = (u_1,\ldots,u_k,v_1,\ldots v_k)$ in step 2 of the algorithm when $w(u_k,v_1) = 1$. Since all $u_1,\ldots,u_k$ have the same status as well as $v_1,\ldots,v_k$, and either u_k or v_1 are faulty, k faulty units have been isolated to within t_q of length 2k. Hence, k is added to f. Furthermore, t_q is added to L_3 so that f faulty units are isolated to within a set of 2f units. It is important to understand that if both u_k and v_1 are faulty, at

least f+k units are faulty.

Theorem 2 The algorithm correctly identifies the status of all units in the system.

Proof: Because of space limitations, simple details are omitted from the proof. First, it must be shown that the tester r, chosen in step (4) is chosen from only fault-free units; that all t_i for which $|t_i| > t-f$ are fault-free. If t_i contained faulty units, then because of observations (1)-(3) there would be more than t faulty units. It is not difficult to see that the status of units tested in step (5) were correctly determined.

It remains to be shown that units tested in step (7) and (8) were accurately diagnosed. Note that t-1 faulty units were identified and that in step (7) we are only concerned with units in L_3. Consider $(u_1,\ldots,u_j,v_1,\ldots,v_j)$ and $w(r,u_j) = 1$ (the other case is trivial) and $j > 1$. All $v_1 \ldots v_j$ must be fault-free because $f + j > t$.

In step (8), we are concerned with L_4 and outside of L_4 t-1 faulty units were identified. Hence, there can be at most one faulty unit in L_4. Consider $w(u,v)$ for u and v in L_4. If $w(u,v) = 0$, then both units must be fault-free. On the other hand, if $w(u,v) = 1$, either u or v is the t^{th} faulty unit and all other units in L_4 are fault-free.

Theorem 3 Let T_n denote the number of tests required by this algorithm. $T_n \leqslant n + t - 3$ test results.

Proof: In Algorithm 1, (except when $f \geqslant t-1$) every unit is tested exactly once except $E^1 = \{v_1 \varepsilon | V\ w(u_k,v_1)=1$ in step 2$\}$ which are tested twice and $E^0 = \{u_1 \varepsilon V \mid t_i = (u_1,\ldots,u_k) \varepsilon L_1 \cup L_2$ and $|t_i| > t-f\}$, which are not tested. The number of units in E^0 corresponds to the number of test results of zero encountered the last time step (2) is executed. Thus $T_n \leqslant n + |E^1| - |E^0|$. If $|E_1| = t$, then the algorithm requires at most $\lceil n/2 \rceil + t$ test results. If $|E_1| = t-1$, the technique used in steps (7) and (8) can be utilized and for all $t_i \varepsilon L_1 \cup L_2$, $t_i = (u_1,u_2)$. In this case, no more than $\lfloor n/2 \rfloor + t-1 + \lfloor (t-1)/2 \rfloor + 1$ test results are required, which is clearly less than n+t-3. Lastly, consider $|E^1| = t-2$. For this choice of $|E^1|$, $|E^0| \geqslant 1$. Hence, the upper bound of $n + t - 3$ is established.

This upper bound is better than the best upper bound for adaptive diagnosis of systems under the PMC model [6], which is $n + t - 1$. However, we believe that $n + t - 3$ is not a best upper bound for T_n and conjecture that $T_n \leqslant \lfloor n/2 \rfloor + 2(t-2) + \lceil (\lceil n/2 \rceil - (t-2))/2 \rceil$.

THE SECOND MODEL

The BGM model takes advantage of the fact that a tester's evaluation of another unit involves many tests and the comparison of the results the tester expects with the answers the tested unit provides. It is assummed that two faulty units will disagree on test results. However, it is unrealistically assumed that a faulty unit and a fault-free unit may agree. The second model we propose overcomes this shortcoming. Thus, if one of the two units involved in a test is faulty, the tester will evaluate the tested unit as faulty. This means that in our model a test result of zero implies that both tester and tested unit are fault-free, unlike the BGM model in which only the tested unit must be fault-free. The necessary and sufficient conditions for t-diagnosability are described in the following theorem.

Theorem 4 Let G be the digraph representing a system S of n units under the second model. Let G'(V,E) be the underlying undirected graph. S is t-diagnosable if and only if (1) $n \geq t + 2$, (2) $|\Gamma(u)| \geq t$ for all $u \in V$, and (3) if for $(u,v) \in E$, $|\Gamma(u)| = |\Gamma(v)| = t$, and for all $w \in \Gamma(u) - \Gamma(v)$, $\Gamma(w) = \Gamma(v)$ and for all $w \in \Gamma(v) - \Gamma(u)$, $\Gamma(w) = \Gamma(u)$, then $\Gamma(V - (\Gamma(u) \cup \Gamma(v))) \not\supseteq \Gamma(u) \cap \Gamma(v)$.

Proof: The proof of the necessity of conditions (1) and (2) is similar to the proof needed in Theorem 1 and will not be shown. The necessity of condition (3) can be shown by violating it and noting that $(\Gamma(u) \cap \Gamma(v)) \cup (\Gamma(u) - \Gamma(v))$ and $(\Gamma(u) \cap \Gamma(v)) \cup (\Gamma(v) - \Gamma(u))$ are two distinct consistent fault sets of cardinality $\leq t$. It is important to note that for all (u,v) in E, $w(u,v) = w(v,u)$. In this model, X is a consistent fault set if and only if for all $u \in X$, $w(u,v) = 1$ and for all $\{u,v\}$ in V-X, $w(u,v) = 0$.

To prove the sufficiency, let conditions (1)-(3) be satisfied and assume that S is not t-diagnosable. Then, for a given sequence of test results, there exist two distinct consistent fault sets F_1 and F_2 with cardinality $\leq t$. Let $F_{12} = F_1 \cap F_2$, $Z_1 = F_1 - F_{12}$, $Z_2 = F_2 - F_{12}$, and $X = V - (F_1 \cup F_2)$. Note that for all $\{u,v\}$ in Z_1 and for all $\{u,v\}$ in Z_2 that $(u,v) \notin E$. Hence, $|\Gamma(Z_1)| \geq t$ and $|\Gamma(Z_2)| \geq t$. Furthermore, since $X \cap \Gamma(Z_1 \cup Z_2) = \emptyset$, $\Gamma(Z_1) \subseteq F_2$ and $\Gamma(Z_2) \subseteq F_1$. Thus, $\Gamma(Z_1) = F_2$ and $|\Gamma(Z_1)| = |F_2| = t$, and $\Gamma(Z_2) = F_1$ and $|\Gamma(Z_2)| = |F_1| = t$. Also, for all $u \in Z_1$, $\Gamma(u) = F_2$ and for all $v \in Z_2$, $\Gamma(v) = F_1$. It is not difficult to see that for all u in Z_1 and v in Z_2, $\Gamma(v) - \Gamma(u) = Z_1$ and $\Gamma(u) - \Gamma(v) = Z_2$. Furthermore, for all w in Z_1, $\Gamma(w) = \Gamma(u)$ and similarly for all w in Z_2, $\Gamma(w) = \Gamma(v)$. Thus the first part of condition (3) is satisfied, namely: If for $(u,v) \in E$,

$|\Gamma(u)| = |\Gamma(v)| = t$, and for all w in $\Gamma(u) - \Gamma(v)$, $\Gamma(w) = \Gamma(v)$ and for all w in $\Gamma(v) - \Gamma(u)$, $\Gamma(w) = \Gamma(u)$. Hence, $\Gamma(V - (\Gamma(u) \cup \Gamma(v))) \not\subseteq \Gamma(u) \cap \Gamma(v)$, which in other terms is $\Gamma(X) \not\subseteq F_{12}$. This along with $\Gamma(Z_1 \cup Z_2) \cap X = \emptyset$ implies that $\Gamma(X) = \emptyset$. But this is a contradiction.

Condition (3) of the characterization theorem is also difficult to verify for the second model. Hence, we present the following corollary, without proof.

Corollary: Let G be the digraph representing system S of n units under the second model. Let G'(V,E) be the underlying undirected graph. If (1) $n \geqslant t + 2$ and (2) $k(G') \geqslant t$, then S is t-diagnosable.

AN OPTIMAL ADAPTIVE DIAGNOSIS ALGORITHM

We shall use Hakimi and Nakajima's adaptive algorithm [5] for the identification of one fault-free unit (in the BGM model) as the basis for our system diagnosis algorithm. We shall also show that the upper bound on the number of tests needed for system diagnosis is the best possible. The following algorithm determines the status of every unit in the system.

Algorithm 2

0. Given system S with set V of n units. We shall form a graph, G(V,E), in which E (initially empty) represents the tests that were performed. As before, the edge-weights correspond to the actual test results. Let $m = n-t-1$ and $q = (n-1)/m$. Arbitrarily choose some unit r and let $V' = V-\{r\}$. Define $V_1, V_2, \ldots, V_m$ to be a partition of V' into m disjoint subsets: Let the first $n-1-m\lfloor q \rfloor$ be of size $\lceil q \rceil$ and let the remaining subsets be of size $\lfloor q \rfloor$.

1. For each set V_i as i goes from m down to 1, seek w(u,v) for each {u,v} in V_i and let $E \leftarrow E \cup \{(u,v)\}$. If $w(u,v) = 0$, then go to 4, {u,v} are fault-free.

2. r is fault-free.

3. Note that all test results were one. Consequently, there is at most one fault-free unit in each of the m subsets. Since there must be at least m fault-free units in V', each V_i contains exactly one fault-free unit. Thus, for each V_i, seek w(r,v) for all $v \in V_i$ until either (1) $w(u,v) = 0$, which implies that V_i-v are faulty, or (2) all but one unit $x \in V_i$ has been tested by r, which implies that x is fault-free. Stop.

4. Note that all but the last test result was zero. This implies that all $\Gamma\{u,v\}$ in G are faulty. Seek w(u,z) for all $z \in V - \Gamma\{u,v\}$. Stop.

Steps 0-2 comprise the Hakimi-Nakajima algorithm, which was proven to be optimal for the identification of one fault-free unit requiring at most $T_1 = (2n-2-m\lfloor q \rfloor -m)(\lfloor q \rfloor /2)$ tests. It is not difficult to see that Algorithm 2 requires at most $T_1 + n - \text{MIN}\,\{m+1, \lceil q \rceil\}$ tests [5]. Let T_n denote the number of tests done in identifying the status of every unit in the system. Next, we shall prove that the algorithm is optimal in terms of the worst case number of tests.

Theorem 5 For the worst case $T_1 + n - \text{MIN}\,\{m+1, \lfloor q \rfloor\}$ tests are needed for system diagnosis.

Proof: Consider the following plausible sequence of test results.

Stage 1: $w(u,v) = 1$ until m (undirected) cliques are formed among n-1 vertices. Denote $C_1, C_2, \ldots, C_m$ as the vertices in the m cliques, where $|C_1| \leqslant |C_2| \leqslant \ldots \leqslant |C_m|$. Let C_0 denote the nth unit, i.e., $V - \bigcup_{i=1}^{m} C_i$. Let $F = \{u \mid u \in C_i \text{ for some } 1 \leqslant i \leqslant m \text{ and } \Gamma(u) \cap C_j = C_j \text{ for some } i \neq j\} \cup \Gamma(C_o)$ (F contains only faulty units).

Stage 2: $w(u,v) = 1$ unless $u \in C_i$ and $v \in C_j$ $(i \neq j)$ such that for all $x \in C_i$, $\Gamma(x) \cap C_j - F = C_j - F$.

Note that since all tests among units in the same clique have result one, there is at most one fault-free unit in each of the m+1 cliques. Since there must be at least m+1 fault-free units, each clique $C_0, \ldots, C_m$ contains exactly one fault-free unit. In the first stage, $\sum_{i=1}^{m}(1/2)|C_i|(|C_i|-1)$ tests were done in forming the cliques, not counting the edges (tests) between cliques. In the second stage, at least $\sum_{i=1}^{m}|C_j-F|-1$ tests are done in isolating the fault-free units. The -1 term is included because if all but one unit in a clique is known to be faulty, the remaining unit must be fault-free and need not be tested. Furthermore, note that for each element of F, for some j, $|C_j| \geqslant 1$ tests were done, which were not counted in forming the cliques. Using this fact, that $\sum_{i=1}^{m}|C_i| = n-1$, and that no unit in F is in more than one clique, we have

$$T_n \geqslant \sum_{i=1}^{m}(1/2)|C_i|(|C_i|-1) + \sum_{i=1}^{m}|C_i-F|-1 + \sum_{i=1}^{|F|}|C_{ji}| \geqslant$$

$$\sum_{i=1}^{m}(1/2)|C_i|(|C_i|-1) + n-1-m.$$

In order to minimize T_n, $\sum_{i=1}^{m}|C_i|^2$ must be minimized. This term is minimized when the cardinalities of the C_i's differ by at most 1. This is accomplished only when $\sum_{i=1}^{m}(1/2)|C_i|(|C_i|-1) = T_1$. Hence, we have shown that $T_n \geqslant T_1 + n-1-m$.

Next to show that $T_n \geqslant T_1 + n-1 - \lceil (n-1)/m \rceil$, consider the same scenario as above but with the following changes: The last test result, $w(u,v)$ in stage 1 is zero and add $\Gamma\{u,v\}$ to set F. At least

$\sum_{i=1}^{m}(1/2)|C_i|(|C_i|-1)$ tests were done in stage 1. Because two fault-free units, {u,v}, were discovered in some C_j, the status of $\Gamma\{u,v\}$ was immediately determined. In addition, we need not have at least one fault-free unit in each clique, and thus one test for each V_i cannot be eliminated. Hence for stage 2, $\sum_{i=1}^{m}|C_i-F| - C_j$ tests are performed. But as we noted above, for each unit in F, at least one test was done which was not counted in forming the cliques. Thus, $T_n \geqslant \sum_{i=1}^{m}(1/2)|C_i|(|C_i|-1) + \sum_{i=1}^{m}|C_i| - |C_j|$. It is not difficult to see that the preceding expression is minimized when $\sum_{i=1}^{m}(1/2)|C_i|(|C_i|-1) = T_1$. Thus, $T_n \geqslant T_1 + n-1 - \lceil q \rceil$ and we have proven the desired result.

REFERENCES

[1] F.P. Preparata, G. Metze, and R.T. Chien, "On the Connection Assignment Problem of Diagnosable Systems", IEEE Trans. Comput., Vol. C-16, pp. 848-854, December, 1967.

[2] F. Barsi, F. Grandoni, and P. Maestrini, "A Theory of Diagnosability without Repair", IEEE Trans. Comput., Vol. C-25, pp. 585-593, June, 1976.

[3] K. Nakajima, "A New Approach to System Diagnosis", in Proc. 19th Annual Allerton Conference on Communications, Control, and Computing, September, 1981.

[4] S.L. Hakimi and E.F. Schmeichel, "An Adaptive Algorithm for System-level Diagnosis", to appear in Journal of Algorithms.

[5] S.L. Hakimi and K. Nakajima, "Adaptive Diagnosis: A New Theorey of t-fault-diagnosable Systems", to appear in IEEE Trans. Comput..

[6] P.M. Blecher, "On a Logical Problem", Discrete Math, Vol 43, pp. 107-110, January, 1983.

A PARTIAL ORDERING FOR SYSTEM–LEVEL FAULT MODELS

Bruce L. Havlicsek
Westinghouse Elec. Corp.
P.O. Box 746 - MS 1615
Baltimore, MD 21203

G. G. L. Meyer
Elec. Engr. & Comp. Sci. Dept.
The Johns Hopkins University
Baltimore, MD 21218

ABSTRACT

We show that a partial ordering of fault models exists that is useful for relating the diagnostic properties of these models. We present a new abstract fault model structure that introduces the concept of internal syndromes and use this structure to define and construct the order relation.

1. INTRODUCTION

Since the presentation of the Preparata, Metze and Chien fault model [PRE67] in 1967, many system-level models for fault analysis have been developed [BAR76, HOL81, ADH77, MAH76, RUS75a] or rediscovered [SOG64, TOI82]. It is now clear that for these approaches to successfully aid the design of diagnosable systems and the development of efficient fault diagnosing algorithms, a basic understanding of the assumptions and properties of each class of fault models is necessary.

For this reason, we have defined a common fault model structure that permits the representation of all these existing approaches and have identified the common structural properties of these models. This fault model structure has been motivated by the work by Kime [KIM70, KIM79] but differs by the addition of internal syndromes. Internal syndromes permit the precise description of hypotheses in terms of internal properties that may not be observable.

By using internal syndromes, we can approach fault analysis in a manner that clearly reflects one's concept of the fault model structure. In particular, the totally morphic property is identified as the common structural property for all fault models referenced in this paper (except [ADH77] which has few structural constraints). Using the totally morphic concept, we identify a partial ordering for the classes of fault models defined by the works referenced in this paper. This ordering is useful in describing the performance of a diagnostic algorithm developed for one class of fault models if it is applied to another class. (The proof of Lemmas 4.1, 5.1 and 5.2 can be found in [HAV82].)

2. ABSTRACT FAULT MODEL

Let $E = \{f_1, f_2, \cdots, f_n\}$, be the set of elementary fault situations that are identified from a given description of a system. These fault situations are elementary in the sense that there is no need to isolate failures more precisely. Let A be the family of all subsets of E. Thus, the empty element of A, denoted by F_0 represents the nonfaulty situation and all the other elements of A represent all possible fault situations. Associated with the fault model is a testing procedure consisting of a set of p tests, $T = \{t_1, t_2, \cdots, t_p\}$, $p \geqslant 1$. The internal outcome of test t_j is denoted by a variable x_j that takes values in the set Z that may be either the set $\{0,1\}$ or the set $\{0,1,2\}$. A test that produces internal outcome "0" is said to *pass*, a test that produces internal outcome "1" is said to *fail* and a test that produces internal outcome "2" is said to be *invalid* [RUS75a]. The internal test outcome "0" always represents the behavior of each test in the presence of fault situation F_0. The set of internal test outcomes for a single application of all p tests is represented by the p-tuple $x = (x_1, x_2, \cdots, x_p)$. This vector of internal test outcomes is called an *internal syndrome* of the fault model and the space of all internal syndromes is denoted by Z^p. A given testing procedure defines a unique point-to-set *fault situation - internal syndrome (FSIS) map,* $G(.)$, from the domain A of fault situations to collections of non-empty subsets in the range Z^p of all possible internal syndromes. Let $Y = \{0,1\}$ be the set of possible observations of internal test outcomes. Thus, only pass and fail test outcomes are observable where, again, "0" denotes pass and "1" denotes fail. The observation process is represented by the point-to-set map, $H(.)$, from Z into all the non-empty subsets of Y. $H(.)$ is defined such that $H(0) = 0$, $H(1) = 1$ and, when "2" is in Z, $H(2) = \{0,1\}$. The observation of a internal syndrome is represented by the map $H(x) = (H(x_1), H(x_2), \cdots, H(x_p))$, where x is a syndrome in Z^p.

A fault model S is characterized by the quintuple $S = (A, Z^p, Y^p, G(.), H(.))$, where

$$A \xrightarrow{G(.)} Z^p \xrightarrow{H(.)} Y^p.$$

When $Z = \{0, 1\}$, the abstract fault model corresponds to typical fault models since the internal and observable representations for syndromes are identical. However, by letting $Z = \{0,1,2\}$, the internal representation can differ from what can actually be observed and common properties of different modeling approaches can be clearly identified.

3. TOTALLY MORPHIC FAULT MODELS

If a fault model is such that the multiple fault internal syndromes can be obtained from the internal syndromes of their elementary fault components. the model is said to be *morphic.* The term "morphic" was first used in reference to system-level fault models defined by Russell and Kime [RUS75a]; we have used it in a more general sense to refer to a variety of relationships between internal syndromes of elementary and multiple fault situations. The presence of morphism reduces the complexity of determining the FSIS table. One should note that graphical fault models [BAR76, HOL81, MAH76, PRE67, RUS75a, SOG64] have reduced their complexity in this way — the graph is actual a description of only the elementary fault situation and internal syndrome relation.

In this paper we examine the concept of *total morphism,* where total morphism is characterized by an associative and commutative binary operation between internal test outcomes called a *morphic map.*

Definition 3.1: A *morphic map* is an associative and commutative binary operation $*$ on the set Z of internal test outcomes. □

If $a = (a_1, a_2, \cdots, a_p)$ and $b = (b_1, b_2, \cdots, b_p)$ are internal syndromes in Z^p; then $a*b$ is the internal syndrome defined by

$$a*b = (a_1*b_1, a_2*b_2, \cdots, a_p*b_p).$$

If Q and R are subsets of internal syndromes in Z^p; then $Q*R$ is the subset of internal syndromes defined by

$$Q*R = \{ a*b \mid (a,b) \in Q \times R \}.$$

Definition 3.2: A fault model S is totally morphic if and only if at least one morphic map $*$ exists so that for every F_a and F_b in A,

$$G(F_a \cup F_b) = G(F_a) * G(F_b). \quad \square$$

The definition of a totally morphic fault model depends only on the internal representation of the model — A, Z^p, and $G(.)$ — and requires that the internal syndromes of the union of any two fault situations be determined by the internal syndromes of the individual fault situations and a morphic map.

Note that if a fault model is totally morphic, then there always exists at least one morphic map $*$ so that Definition 2 is satisfied and

$$a * a = 0 * a = a * 0 = a$$

for all internal test outcomes a in Z. Accordingly, for a given set Z, the number of morphic maps that need to be considered is small.

4. PASS/FAIL TEST OUTCOMES

In this section, totally morphic fault models in which $Z = \{0,1\}$ are examined. In that case, all totally morphic fault models that possess at least one fault situation that can be distinguished from F_0 have the same morphic map.

Lemma 4.1: Let S be a totally morphic fault model. If (i) $Z = \{0,1\}$ and (ii) at least one fault situation F exists such that $(0, \cdots, 0)$ is not a syndrome in $G(F)$, then only the morphic map $*_1$ corresponds to this fault model, where $*_1$ is defined in the following table.

$*_1$	0	1
0	0	1
1	1	1

Lemma 4.1 implies that all non-trivial, totally morphic models having only pass and fail test internal outcomes have the same morphic map — the comparison of such models requires only the part of the FSIS table corresponding to elementary fault situations. One should note that the earliest known graphical fault model [SOG64] satisfies the assumptions of Lemma 4.1.

5. PASS/FAIL/INVALID TEST OUTCOMES

Several existing models [MAH7, PRE67, RUS75a] are not totally morphic when the set Z is restricted to $\{0,1\}$. It is therefore logical to examine what happens when $\|Z\|$ is increased by one. In this case, the internal test outcome "2" is added to represent the internal knowledge that the behavior of a test is uncertain or invalid.

Lemma 5.1: Let S be a totally morphic fault model. If (i) $Z = \{0,1,2\}$ and (ii) a test t_k and two fault situations F_a and F_b exist such that $G(F_a)_k = 1$ and $G(F_b)_k = 2$, then either the morphic map $*_2$ or the morphic map $*_3$ correspond to this fault model, where $*_2$ is defined by:

$*_2$	0	1	2
0	0	1	2
1	1	1	1
2	2	1	2

and $*_3$ is defined by:

$*_3$	0	1	2
0	0	1	2
1	1	1	2
2	2	2	2

The morphic maps $*_2$ and $*_3$ are essentially identical except that the roles of the argument values "1" and "2" are reversed. Consequently, totally morphic fault models for which $Z = \{0,1,2\}$ are only distinguished by the values of $H(1)$ and $H(2)$ — the observable value of "1" and "2".

It is clear that if a fault model is totally morphic with $Z = \{0,1\}$ (Lemma 4.1), then it is also totally morphic with $Z = \{0,1,2\}$ (Lemma 5.1), where the test outcome "2" never occurs. Thus, totally morphic models with pass/fail test outcomes are particular cases of totally morphic models with pass/fail/invalid test outcomes. It is interesting to note, however, that if a model with pass/fail/invalid test outcomes is totally morphic with morphic map $*_2$, then a fault model exists that is totally morphic with respect to $*_1$ and has identical observable syndromes.

Lemma 5.2: A fault model $S_2 = (\ A,\ \{0,1,2\}^p,\ \{0,1\}^p,\ G_2(.),\ H(.)\ )$ is totally morphic with respect to the morphic map $*_2$ if and only if a fault model $S_1 = (\ A,\ \{0,1\}^p,\ \{0,1\}^p,\ G_1(.),\ H(.)\ )$ that is totally morphic with respect to the morphic map $*_1$ exists such that

$$G_1(F) = H(G_2(F))$$

for every fault situation F in A. □

The consequence of Lemma 5.2 is that using only the two morphic maps, $*_2$ and $*_3$, one can represent any graphical fault model referenced in this paper as a totally morphic fault model. Without the use of internal syndromes — in particular, the internal test outcome "2" — it is not possible to represent some of these models [MAH76, PRE67, RUS75a] as totally morphic.

6. APPLICATION EXAMPLES

The abstract fault model defined in this paper is useful for representing, comparing, and classifying new or existing fault models, since any model having only pass and fail observable syndromes has at least one representation. Our model provides a framework to define classes of totally morphic fault models and to 1) determine the structure of the elementary fault situation internal syndromes that provides diagnosability, 2) develop diagnostic algorithms for each class of models, and, more importantly, 3) apply diagnostic properties and algorithms from one class of fault models to another class.

The initial step of this analysis that is presented in this paper is to examine previously defined fault models having totally morphic representations. In this section, the abstract model is applied to these fault models to demonstrate that a partial ordering exists that relates their properties. The fault model representation discussed in Section 5 is used since all examples have a totally morphic representation.

6.1 BGM FAULT MODEL

The Barsi, Grandoni and Maestrini (BGM) [BAR76] model assumes that a system is partitioned into n subsystem units $\{u_1, \cdots, u_n\}$. A unit may be either faulty or nonfaulty; therefore, a fault situation is completely described by the subset of faulty units. When this subset is empty, denoted by F_0, the system is nonfaulty. The units test each other according to a testing interconnection assignment

$$I = \{\ (i_k, j_k)\ |\ 1 \leqslant k \leqslant p,\ u_{ik} \text{ tests } u_{jk}\ \}.$$

Associated with each pair (i_k, j_k) in I is an observable test outcome variable y_k that takes the value "0" when both units u_{ik} and u_{jk} are nonfaulty, "1" when unit u_{jk} is faulty, and either "0" or "1" otherwise. An observable syndrome y is the ordered vector of all y_k for all (i_k, j_k) in I.

Abstract Representation:

(i) $E = \{f_1, \cdots, f_n\}$, where f_i denotes that unit u_i is faulty.

(ii) $T = \{t_1, \cdots, t_p\}$, where t_k denotes test (i_k, j_k) in I.

(iii) $G(.)\!:\!A \rightarrow Z^p$ is a point-to-point map such that if F is in A, then $G(F) = (x_1, \cdots, x_p)$ where

$$x_k = \begin{cases} 0, & \text{if neither } f_{ik} \text{ nor } f_{jk} \text{ is in } F, \\ 1, & \text{if } f_{jk} \text{ is in } F, \\ 2, & \text{if } f_{ik} \text{ is in } F, \text{ and } f_{jk} \text{ is not in } F \end{cases}$$

It is easily verified that of the above model represents the fault model of Barsi, Grandoni and Maestrini. The FSIS table is a point-to-point map and the model is totally morphic with respect to the morphic map $*_2$. One should recall that $y_k = H(x_k)$ and that the map $H(.)$ is a point-to-set map. Hence, in this representation the ambiguous nature of the fault situations has been restricted to the observation process. The internal syndromes of the FSIS table represent well defined internal properties that cannot be directly observed.

6.1 PMC FAULT MODEL

The Preparata, Metze, and Chien (PMC) fault model assumes that a system is partitioned into n subsystem units $\{u_1, \cdots, u_n\}$. A unit may be either faulty or nonfaulty, therefore a fault situation is completely described by the subset of units that are faulty. When this subset is empty, denoted by F_0, the system is nonfaulty. The units test each other according to a testing interconnection assignment

$$I = \{ (i_k, j_k) \mid 1 \leqslant k \leqslant p, u_{ik} \text{ tests } u_{jk} \}.$$

Associated with each pair (i_k, j_k) in I is an observable test outcome variable y_k that takes the value "0" when both units u_{ik} and u_{jk} are nonfaulty, "1" when unit u_{ik} is nonfaulty and u_{jk} is faulty, and either "0" or "1" otherwise. An observable syndrome y is the ordered vector of all y_k for all (i_k, j_k) in I.

Abstract Representation:
(i) $E = \{f_1, \cdots, f_n\}$, where f_i denotes that unit u_i is faulty.
(ii) $T = \{t_1, \cdots, t_p\}$, where t_k denotes test (i_k, j_k) in I.
(iii) $G(.): A_n \rightarrow Z^p$ is a point-to-point map such that if F is in A_n, then $G(F) = (x_1, \cdots, x_p)$ where

$$x_k = \begin{cases} 0, & \text{if neither } f_{ik} \text{ nor } f_{jk} \text{ is in } F, \\ 1, & \text{if } f_{ik} \text{ is not in } F, \text{ and } f_{jk} \text{ is in } F,. \\ 2, & \text{if } f_{ik} \text{ is in } F \end{cases}$$

We see that the internal representation of the PMC fault model differs from that of the BGM fault model only for multiple fault situations. This is consistent with the fact that the PMC fault model is totally morphic with respect to the morphic map $*_3$. Hence, some occurrences of the internal test outcome "1" in the BGM fault model, take the value "2" in the PMC fault model.

6.3 SOG FAULT MODEL

The modeling approach of Sogomonyan (SOG) [SOG64] assumes that a system is partitioned into n subsystem units. Each unit u_j has a set of input variables, $B_j = \{b_{j1}, b_{j2}, \cdots, b_{j\|B_j\|}\}$, and a single output variable, a_j. Modules are interconnected according to an interconnection assignment

$$I = \{(i,j,k) | \text{output of unit } i \text{ is connected to input } k \text{ of unit } j\}.$$

If for some $1 \leqslant j \leqslant n$ and $1 \leqslant k \leqslant \|B_j\|$, (i,j,k) is not in I for all $1 \leqslant i \leqslant n$, then input k of unit j is an "external" input and $b_{jk} = 0$ by assumption of the model. An interconnection assignment is *proper* if and only if (i,j,k) in I implies that for all fault situations, $a_i = b_{jk}$. Thus, knowledge of the values of the variables a_i for all $1 \leqslant i \leqslant n$ implies the values of all variables b_{jk} for all $1 \leqslant j \leqslant n$ and $1 \leqslant k \leqslant \|B_j\|$. Given a proper interconnection assignment, an output variable a_j takes the value

"0" when $b_{jk} = 0$ for all $1 \leqslant k \leqslant \| B_j \|$ and unit u_j is nonfaulty, and is "1" otherwise. An observable syndrome $y = (a_{q1}, a_{q2}, \cdots, a_{qp})$ is the ordered vector of p output variables defined by the set
$Q = \{q1, q2, \cdots, qp\} \subseteq \{1, 2, \cdots, n\}$.

Abstract Representation:
(i) $E = \{f_1, \cdots, f_n\}$, where f_i denotes that unit u_i is faulty.
(ii) $T = \{t_1, \cdots, t_p\}$, where test t_i consists of observing the output of unit qi, $qi \in Q$.
(iii) $G(.): A_n \rightarrow Z^p$ is a point-to-point map such that if F is in A_n, then $(a_{q1}, \cdots, a_{qp}) = G(F)$, $qi \in Q$.

In this case, the FSIS table is a point-to-point map and the observable syndromes are identical to the internal syndromes since the internal value "2" never occurs. It should be noted that the syndromes corresponding to this fault model are identical to those defined by other graphical fault models described by Conti [CON74] and Toida [TOI82].

6.4 RK FAULT MODEL

The modeling approach of Russell and Kime (RK) [RUS75a] uses a set of n elementary fault situations $E = \{f_1, \cdots, f_n\}$ and a set of p tests $T = \{t_1, \cdots, t_p\}$. Associated with every test t_k in T is an elementary fault situation $C(t_k)$ in E and a fault situation, $I(t_k)$, in $E - \{C(t_k)\}$. Also associated with each test is an observable test outcome variable y_k. An observable syndrome y is the ordered vector of all test outcome variables y_k, $1 \leqslant k \leqslant p$. For a fault situation, F, y_k takes the value "0" when $(I(t_k) \cup C(t_k)) \cap F = \phi$, "1" when $I(t_k) \cap F = \phi$ and $C(t_k) \cap F \neq \phi$, and either "0" or "1" otherwise.

Abstract Representation:
(i) $E = \{f_1, \cdots, f_n\}$, where f_i denotes the i^{th} elementary fault.
(ii) $T = \{t_1, \cdots, t_p\}$, where t_k denotes the k^{th} test.
(iii) $G(.): A_n \rightarrow Z^p$ is a point-to-point map such that if F is in A_n, then $G(F) = (x_1, \cdots, x_p)$, where

$$x_k = \begin{cases} 0, \text{ if } (I(t_k) \cup C(t_k)) \cap F = \phi, \\ 1, \text{ if } I(t_k) \cap F = \phi \text{ and } C(t_k) \cap F \neq \phi,. \\ 2, \text{ if } I(t_k) \cap F \neq \phi \end{cases}$$

In this case, the FSIS table is a point-to-point map and the model is totally morphic with respect to the morphic map $*_3$. Recall that $H(x_k) = y_k$ and that $H(.)$ is a point-to-set map.

6.5 HS FAULT MODEL

The modeling approach of Holt and Smith (HS) [HOL81] is similar to that of Russell and Kime, except that $C(t_k)$ may be any fault situation instead of only an elementary fault situation, and the observable test outcome variable y_k takes values according to a different hypothesis. If F is a fault situation, then y_k takes the value "0" when $(I(t_k) \cup C(t_k)) \cap F = \phi$, "1" when $C(t_k) \cap F \neq \phi$, and either "0" or "1" otherwise.

Abstract Representation:
(i) $E = \{f_1, \cdots, f_n\}$, where f_i denotes the i^{th} elementary fault.
(ii) $T = \{t_1, \cdots, t_p\}$, where t_k denotes the k^{th} test.
(iii) $G(.): A_n \rightarrow Z^p$ is a point-to-point map such that if F is in A_n, then $G(F) = (x_1, \cdots, x_p)$, where

$$x_k = \begin{cases} 0, \text{ if } (I(t_k) \cup C(t_k)) \cap F = \phi, \\ 1, \text{ if } C(t_k) \cap F \neq \phi, \\ 2, \text{ if } I(t_k) \cap F \neq \phi \text{ and } C(t_k) \cap F = \phi \end{cases}$$

In this case, the FSIS table is a point-to-point map and the model is totally morphic with respect to the morphic map $*_2$. Again, $H(x_k) = y_k$ and $H(.)$ is a point-to-set map.

7. PARTIAL ORDERING OF MORPHIC FAULT MODELS

The purpose of this research is to allow us to better understand system-level fault models and to be able to transfer information about the properties and algorithms of one class of fault models to another class. For this reason we define the following order relation between fault models based on their observable syndromes.

Definition 7.1: Let $S_a = (\ A,\ Z_a^p,\ Y^p,\ G_a(.),\ H(.)\ )$ and $S_b = (\ A,\ Z_b^p,\ Y^p,\ G_b(.),\ H(.)\ )$ be fault models. (Note that A and Y^p are common.) S_a is *ordered less than or equal to* S_b, denoted $S_a \leqslant S_b$ or $S_b \geqslant S_a$, if and only if

$$H(G_a(F)) \subseteq H(G_b(F))$$

for every F in A. □

This definition implies that all observable syndromes of S_a are contained in S_b; therefore, the diagnosability properties of S_b must also be true for S_a. This definition is extended to classes of fault models as follows.

Definition 7.2: Let C_a and C_b be classes of fault models. C_a is *ordered less than or equal to* C_b if and only if for every fault model $S_a = (\ A,\ Z_a^p,\ Y^p,\ G_a(.),\ H(.)\ )$ in C_a, a fault model $S_b = (\ A,\ Z_b^p,\ Y^p,\ G_b(.),\ H(.)\ )$ in C_b exists such that $S_a \leqslant S_b$. □

This definition implies a partial order relation exists for the previously described fault model classes. The most important aspect of this ordering is that any diagnostic algorithm for fault models in class C_b can be immediately applied to the models in class C_a. A secondary aspect of the ordering is that given a fault model in class C_b, one can apply diagnostic algorithms from class C_a whenever the syndromes to be decoded are common to that class as well. This is particularly interesting in the case where the diagnostic algorithms for class C_a are much simplier than those of class C_b. We note that this order relation does not require the use of internal syndromes. However, for totally morphic fault models, the use of internal syndromes leads to a simple construction of this ordering. The following relation on the set Z forms the basis of our construction approach.

Definition 7.3: Let $\leqslant$ be a relation on the set $Z = \{0,1,2\}$ defined as follows

			y	
$x \leqslant y$		0	1	2
x	0	$\leqslant$		$\leqslant$
	1		$\leqslant$	$\leqslant$
	2			$\leqslant$

□

The symbol $\leqslant$ has been used to denote relations on the set of all fault models, the set of all classes of fault models and above on the set of internal test outcomes, $\{0,1,2\}$. In each case, however, the definition that applies is clear according to the context. Using Definition 7.3, the following lemma can be easily verified by checking all cases.

Lemma 7.4: Let w, x, y, z be elements of the set $\{0,1,2\}$. If $w \leqslant x$ and $y \leqslant z$, then all of the following are true.

$$w *_2 y \leqslant x *_2 z$$

$$w *_3 y \leqslant x *_3 z$$

$$w *_2 y \leqslant x *_3 z. \quad \square$$

Lemma 7.4 implies the following sufficient conditions for the ordering of fault models.

Theorem 7.5: Let $S_a = (\ A,\ \{0,1,2\}^p,\ \{0,1\}^p,\ G_a(.),\ H(.)\ )$ and $S_b = (\ A,\ \{0,1,2\}^p,\ \{0,1\}^p,\ G_b(.),\ H(.)\ )$ be totally morphic fault models satisfying Lemma 5.1 and where $G_a(.)$ and $G_b(.)$ are point-to-point maps. If (i) $G_a(f)_k \leqslant G_b(f)_k$ for all f in E and (ii) either S_a and S_b are totally morphic with respect to the same morphic map or S_a is totally morphic with respect to $*_2$ and S_b is totally morphic with respect to $*_3$, then $S_a \leqslant S_b$.

PROOF: Let $F \in A$ and let $X = \{ f_{i1}, \cdots, f_{im} \} \subseteq E$, where $m = \|F\|$ and $\bigcup_{f \in X} f = F$. By Definition 3.2, for every $1 \leqslant k \leqslant p$,

$$G_a(F)_k = G_a(f_{i1})_k *_a \cdots *_a G_a(f_{im})_k$$

and

$$G_b(F)_k = G_b(f_{i1})_k *_b \cdots *_b G_b(f_{im})_k$$

where S_a is totally morphic with respect to $*_a$ and S_b is totally morphic with respect to $*_b$. Since by (i) $G_a(f)_k \leqslant G_b(f)_k$ for every f in E and by (ii) $*_a = *_b$ or $*_a = *_2$ and $*_b = *_3$, then by Lemma 7.4,

$$G_a(f_{i1})_k *_a \cdots *_a G_a(f_{ir})_k \leqslant G_b(f_{i1})_k *_b \cdots *_b G_b(f_{ir})_k$$

for all $1 \leqslant r \leqslant m$. Hence, $G_a(F)_k \leqslant G_b(F)_k$ which implies $H(G_a(F)_k) \subseteq H(G_b(F)_k)$. Therefore, $S_a \leqslant S_b$. $\square$

The use of internal syndromes in our abstract fault model structure allows the internal representation of the fault models to be point-to-point maps. This in turn allows the use of the simple order relation given in Definition 7.3 in Theorem 7.5.

From the examples in Section 6, we see that all these fault models satisfy the conditions of this theorem and thus Theorem 7.5 becomes the basis of a partial ordering on theses classes of fault models. The following theorem shows that condition (i) of Theorem 7.5 is also a necessary condition for the partial ordering to exist.

Theorem 7.6: Let $S_a = (\ A,\ \{0,1,2\}^p,\ \{0,1\}^p,\ G_a(.),\ H(.)\ )$ and $S_b = (\ A,\ \{0,1,2\}^p,\ \{0,1\}^p,\ G_b(.),\ H(.)\ )$ be fault models where $G_a(.)$ and $G_b(.)$ are point-to-point maps. If $S_a \leqslant S_b$, then $G_a(f)_k \leqslant G_b(f)_k$ for all f in E and all $1 \leqslant k \leqslant p$.

PROOF: Assume $f \in E$ and $1 \leqslant k \leqslant p$ exist such that $G_a(f)_k \leqslant G_b(f)_k$ is not true. Then, four possible cases may exist as shown below.

$G_a(f)_k$	$H(G_a(f)_k)$	$G_b(f)_k$	$H(G_b(f)_k)$
0	0	1	1
1	1	0	0
2	{0,1}	0	0
2	{0,1}	1	1

In each case, $H(G_a(f)) \not\subseteq H(G_b(f))$. Hence, S_a is not less than or equal to S_b. $\square$

We now can construct the ordering of the fault models described in Section 6 in two steps. First, construct a partial ordering based on the necessary conditions given in Theorem 7.6, then complete the ordering based on the sufficient conditions of Theorem 7.5. We can add to the ordering the class of all fault models defined by the abstract model structure given in Section 2. This latter class is equivalent to the class of models described by Adham and Friedman [ADH77]. The final ordering is shown graphically in Figure 1.

8. CONCLUSION

We have defined an abstract fault model structure using internal syndromes that allows us to construct a partial ordering relation for classes of fault models. Since this order relates the diagnosability of different fault models, we are developing algorithms that can be applied to multiple classes of fault models and take advantage of the common syndromes described by the ordering relationship.

9. AKNOWLEDGEMENT

This work was supported in part by the Office of Naval Research under Contract N00014-80-C-0772 and in part by Westinghouse Electric Corporation, Integrated Logistics Support Division.

10. REFERENCES

[ADH77] Adham, M. and Friedman A., Digital System Fault Diagnosis, Design Automation and Fault-Tolerant Computing, Vol. 1, No. 2, February 1977, pp. 115-132.

[BAR76] Barsi, F., Grandoni, F. and Maestrini, P., A Theory of Diagnosability of Digital Systems, IEEE Transactions on Computers, Vol. C-25 , June 1976, pp. 585-593.

[CON74] Conti, R.A., A method of Fault Isolation for Large Digital Systems, Proceedings of EASCON 74, 1974.

[HAV82] Havlicsek, B.L. and Meyer, G.G.L., A New System-level Approach to Diagnosability, The Johns Hopkins University, Department of Electrical Engineering and Computer Science, Technical Report JHU-EECS-82-6, 1982.

[HOL81] Holt, C.S. and Smith, J.E., Diagnosis of Systems with Asymmetric Invalidation, IEEE Transactions on Computers, Vol. C-30, No. 9, September 1981, pp. 679-690.

[KIM70] Kime, C.R., An Analysis Model for Digital System Diagnosis, IEEE Transactions on Computers, Vol. C-19, No. 11, November 1970, pp. 1063-1073.

[KIM79] Kime, C.R., An Abstract Model for Digital System Fault Diagnosis, IEEE Transactions on Computers, Vol. C-28, No. 10, October 1979, pp. 754-767.

[MAH76] Maheshwari, S.N. and Hakimi, S.L., On Models for Diagnosable Systems and Probabilistic Fault Diagnosis, IEEE Transactions on Computers, Vol. C-25, No. 3, March 1976, pp. 228-236.

[PRE67] Preparata, F.P., Metze, G. and Chien, R.T., On the Connection Assignment Problem of Diagnosable Systems, IEEE Transactions on Electronic Computers, Vol. EC-16 , December 1967, pp. 848-854.

[RUS75a] Russell, J.D. and Kime, C.R., System Fault Diagnosis: Closure and Diagnosability with Repair, IEEE Transactions on Computers, Vol. C-24, November 1975, pp. 1078-1089.

[SOG64] Sogomonyan, E.S., Monitoring Operability and Finding Failures in Functionally Connected Systems, Automatic and Remote Control, Vol. 25, No. 6, June 1964, pp. 874-882.

[TOI82] Toida, S., A Graph Model for Fault Diagnosis, Journal of Digital Systems, Vol. VI, No. 4, Winter 1982, pp. 345-365.

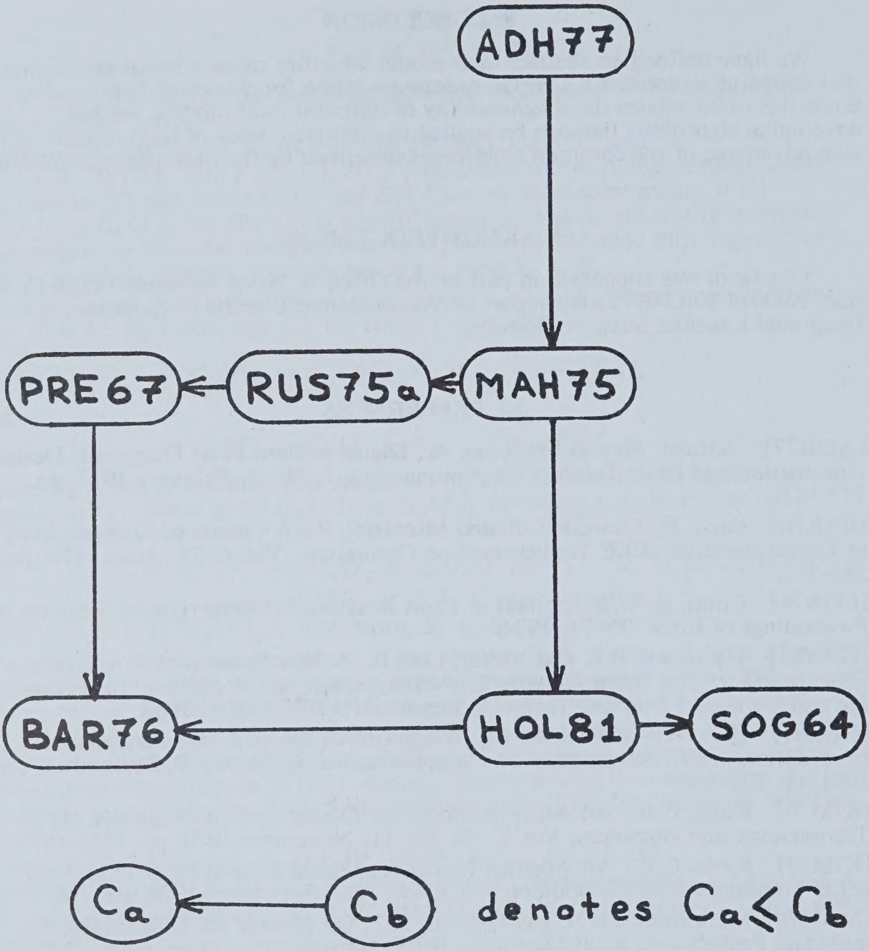

ADH77
PRE67
RUS75a
MAH75
BAR76
HOL81
SOG64
C_a
C_b
denotes $C_a \leq C_b$

Figure 1

ON ADAPTIVE FAULT IDENTIFICATION IN OPTIMAL DIAGNOSABLE SYSTEMS*

KAZUO NAKAJIMA and VIRABRAHMA P. KROTHAPALLI
Department of Electrical Engineering/Computer Science
Texas Tech University
Lubbock, Texas 79409

ABSTRACT

In the earlier work on the adaptive system diagnosis problem, it was assumed that every unit is capable of testing every other unit in the system. Several adaptive diagnosis algorithms were developed for such systems. Recently, two of the algorithms were shown to be applicable to a certain class of "optimal" t-fault diagnosable systems in which every unit is tested by exactly t other units. We show that a variation of the Hakimi-Schmeichel algorithm identifies a fault-free unit for another class of optimal systems. We then present a necessary and sufficient condition for identification of all faulty units after locating a fault-free unit. Finally, we briefly discuss the adaptive fault identification problem for undirected graph models.

1. INTRODUCTION

Consider a system S consisting of a set U of n units $u_1, u_2, \ldots, u_n$ and a set T of m test links $t_1, t_2, \ldots, t_m$, where test link $t_k=(u_i,u_j)$ from unit u_i to unit u_j signifies that unit u_i is capable of testing unit u_j. Thus, the units may be represented as the vertices and the test links as the edges of a digraph G(U,T). The result of the test carried out by unit u_i on unit u_j using test link $t_k=(u_i,u_j)$ is either fault-free or faulty, and it is reliable only if u_i is fault-free. The test result may be represented as the weight $R(t_k) = R(u_i,u_j)$ on the edge $t_k=(u_i,u_j)$ of G, where $R(t_k)=0$ (resp., 1) if it is fault-free (resp., faulty). A system S is defined to be t-diagnosable if, given the test results, all faulty units in S can be identified, provided the number of faulty units does not exceed t. Preparata, Metze and Chien [1] showed that if a system S of n units in the "PMC model" is t-diagnosable then $n \geq 2t+1$ and each unit must be tested by at least t other units in S. Hakimi and Amin [2] then showed that these conditions are in fact sufficient if no two units test each other in S. They also obtained necessary and sufficient conditions for the general case when no such restriction is placed on S. Since then the theory of t-diagnosable systems has rapidly advanced [3-17].

In such a theory, it is explicitly assumed that one first chooses a set of tests, then one seeks the results of these tests, and finally one proceeds to use the test results to identify the faulty units provided that the number of faulty units does not exceed t. Recently, Nakajima [18] suggested an adaptive fault diagnosis approach. Assuming that we have no more than t faulty units, he proposed that one adaptively select the tests and obtain their results until one can identify a fault-free unit. Then one can use this unit as a tester to identify the status of the remaining n-1 units. He showed that one can identify a fault-free unit by using the results of at most t(t+1) tests. Thus, all faulty units can be identified in at most (n-1)+t(t+1) tests.

---

*This work is supported by the Joint Service Electronics Program under ONR Contract N00014-76-C-1136 and the Institute for University Research in Engineering at Texas Tech University.

Hakimi and Nakajima [19,20] exploited this adaptive approach further. In particular, they developed an adaptive algorithm which identifies a fault-free unit in at most 2t-1 tests, provided the number of faulty units does not exceed t. Thus, one needs at most n+2t-2 tests to identify all faulty units. Blecher [21] improved this bound by using a different adaptive procedure. He showed that all faulty units can be identified in at most n+t-1 tests, although 2t-1 tests are needed in the worst case to identify a fault-free unit. He also proved that n+t-1 is the lower bound on the number of tests required to identify all faulty units. Quite recently, Hakimi and Schmeichel [22] discovered an adaptive algorithm which requires at most $2t-B_1(t)$ tests to identify a faulty-free unit, where $B_1(t)$ is the number of 1's in the binary representation of t.

In the application of these adaptive algorithms, it is explicitly assumed that every unit in the system is capable of testing every other unit, that is, there are n(n-1) test links in the system of n units. As mentioned before, however, a system S of n units in which no two units test each other is t-diagnosable if and only if every unit is tested by at least t other units [2]. Preparata et al. [1] proposed a class of "optimal" t-diagnosable systems, called $D_{\delta t}$, in which no two units test each other and each unit is tested by exactly t other units. Chwa and Hakimi [14] proposed another class of such systems, called D(n,t,X). Hakimi and Nakajima [20] showed that their adaptive algorithm identifies a fault-free unit in a $D_{\delta t}$ system. The authors [23] reported that the Hakimi-Schmeichel algorithm is applicable to a $D_{\delta t}$ system.

In this paper, we introduce a new class of optimal t-diagnosable systems, called D*(n,t,X), which is a subclass of D(n,t,X). We show that a variation of the Hakimi-Schmeichel algorithm always identifies a fault-free unit in a D*(n,t,X) system. We then consider the problem of identifying all faulty units after locating a fault-free unit. We present a necessary and sufficient condition for such a purpose. Finally, we briefly discuss the adaptive fault identification problem for the undirected graph model of Chwa and Hakimi [16] and its variations. This problem was recently studied by Nakajima [25].

In Section 2, we define the class D*(n,t,X) of optimal t-diagnosable systems. We present an adaptive diagnosis algorithm which is a variation of the Hakimi-Schmeichel algorithm. We prove that the algorithm always identifies a fault-free unit in a D*(n,t,X) system in at most $2t-B_1(t)$ tests. In Section 3, we present a characterization theorem for identification of all faulty units after locating a fault-free unit. In Section 4, we discuss the application of the adaptive fault diagnosis approach to the Chwa-Hakimi undirected graph model and it variations. We describe some of Nakajima's [25] recent work in this field. Section 5 concludes the paper with some future research problems in the area of adaptive system diagnosis.

2. ADAPTIVE DIAGNOSIS IN D*(n,t,X) SYSTEMS

We begin by defining a new class of "optimal" t-diagnosable systems in which no two units test each other and each unit is tested by exactly t other units. Let U be a set of n units $u_0,u_1,\ldots,u_{n-1}$ and T be a set of nt test links to be defined below. Let X be a set of t integers $x_1,x_2,\ldots,x_t$ with $1\le x_1<x_2<\ldots<x_t<n/2$, and let $a_1 = x_1$ and $a_i = x_i-x_{i-1}$ for $1<i\le t$.

Definition: A system S of n units, represented by a digraph G(U,T), is said to be in D*(n,t,X) if

(1) $(u_i,u_j) \in T$ if and only if $(i-j) \bmod n \in X$, and

(2) $a_i\ge a_{i-1}$ for $1<i\le t$.

We call a member of the class $D^*(n,t,X)$ a $D^*(n,t,X)$ system. When the second condition is omitted, the $D(n,t,X)$ class introduced by Chwa and Hakimi [3] is obtained. Figure 1 illustrates the digraph model of a $D^*(13,3,\{1,2,4\})$ system.

In this section, we show that a variation of the Hakimi-Schmeichel algorithm adaptively identifies a fault-free unit in a $D^*(n,t,X)$ system. Before presenting our version of their algorithm, let us observe the properties of a special type of in-trees which are generated during the execution of the algorithm. Such trees are called binomial trees [24] and illustrated in Fig. 2.

The number of vertices in any binomial tree is of the power of two. The order of a binomial tree is defined by the logarithm (base two) of the number of vertices in it. The root of a binomial tree is the vertex towards which all the incident edges are directed. A binomial tree of order i is recursively obtained from two binomial trees of order i-1 by drawing a directed edge from the root of one tree to the root of the other. Hence, the number of edges in a binomial tree of order i is 2^i-1.

In a weighted binomial tree of order i, if all the edges are of weight zero, then all the 2^i units are faulty whenever the root (unit) is faulty. If all edges except the one between the roots of the two binomial subtrees of order i-1 have zero weights, then there are at least 2^{i-1} faulty units in the tree since at least one of the roots of the subtrees is faulty.

We now present an adaptive algorithm which is a variation of the Hakimi-Schmeichel algorithm [22]. In the following algorithm, f(u) is the number of units that would be identified as faulty from the previous test results, purely as a consequence of the assumption that unit u is faulty; F is the number of faulty units that have been isolated within a set of size 2F; S is a stack of ordered pairs $(f(u_r),r)$, where r is the subscript of the root u_r of a binomial tree which contains $f(u_r)$ units; and |S| denotes the number of elements in S. Let A(u) denote the set of immediate ancestors of unit u with respect to the original digraph G(U,T).

Algorithm 1:

```
Step 0.  Set $F \leftarrow 0$, $f^* \leftarrow 0$, $r \leftarrow 0$, $j \leftarrow x_t$, $M \leftarrow 1$, $S \leftarrow \phi$ and $k \leftarrow 1$.
Step 1.  Seek the result $R(u_j,u_r)$ of test $(u_j,u_r)$ and set $U \leftarrow U - \{u_j,u_r\}$.
         (a)  If $R(u_j,u_r) = 0$, set $M \leftarrow M*2$ and push $(M,r)$ on to S.  If $f^* < M$, set $f^* \leftarrow M$.  GO TO Step 2.
         (b)  If $R(u_j,u_r) = 1$, set $F \leftarrow F + M$.
              If $F \geq t$, STOP.  All units in U and all the units of the trees whose roots are stored in S are fault-free.
              If $S = \phi$, set $r \leftarrow x_k$, $j \leftarrow r + x_t$, $M \leftarrow 1$, $f^* \leftarrow 0$, $k \leftarrow k+1$, and GO TO Step 1.
Step 2.  If $F + f^* > t$, STOP.  Let $(f(u_i),i)$ be the bottom element of S; then $u_i$ is fault-free.
         If $|S| \geq 2$, let $(f(u_p),p)$ and $(f(u_q),q)$ be the first and second top most elements of S.  If $f(u_p) = f(u_q)$, pop these elements out of S, and set $j \leftarrow p$, $r \leftarrow q$ and GO TO Step 1.
         Otherwise (i.e., either $|S| = 1$ or $f(u_p) \neq f(u_q)$), set $M \leftarrow 1$ and set r to be the lowest index $\ell$ such that $x_{k-1} < \ell \leq x_k$ and $u_\ell \in A(u_p) \cap U$, where if $|S| = 1$, $f((u_p),u_p)$ is the only element of S, and set $j \leftarrow r + x_t$, $k \leftarrow k + 1$, and GO TO Step 1.
```

We prove that the above algorithm always identifies a fault-free unit in a $D^*(n,t,X)$ system using 2t+1 units and at most $2t-B_1(t)$ tests. This is done through a series of lemmas. We first note that tests to be conducted at Step 1 are categorized into two types. One is that of tests in which two units participate for the first time. In other words, these tests yield binomial trees of order one. We call them simple tests. The other is the kind in which the roots of two binomial trees of order one or more are involved. We call them compound tests. A simple test is carried out at Step 1 on entry from Step 0 and afterwards from Step 1.b and the last part of Step 2. A compound test is performed when we return to Step 1 from Step 1.b.

A compound test (producing a binomial tree) of order i(>1) contains 2^{i-1} simple tests. When a compound test of order i fails, we know that all of the units in at lest one of the participating binomial trees of order i-1 are faulty; that is, at least 2^{i-1} units are faulty. This is equivalent to the situation in which all 2^{i-1} simple tests contained in the compound test of order i fail. Therefore, it is easy to see that in the above algorithm the parameter F denotes the number of simple tests that have failed, or the least number of faulty units that have been isolated. We also note that the index k denotes the number of simple tests that have been carried out at Step 1 of the algorithm.

We first prove that we can always perform a simple test; in other words, we can always find a unit to be tested in Step 1.b and in the last part of Step 2. In the proofs of the following lemmas, we should keep in mind that the index k never exceeds t, which will be later proven. Since $x_t \geq t$ by the definition of a $D^*(n,t,X)$ system, all the units involved as testers in simple tests are of index $\geq x_t$. Thus, all units to be concerned with in these lemmas are among those whose indices range from 0 to x_t.

Lemma 1: For any unit u_i, $0 \leq i \leq x_{k-1}$, there exists at least one unit u_j such that $u_j \in A(u_i)$ and $x_{k-1} < j \leq x_k$.

The proof of this lemma is easy and thus it is omitted.

Lemma 2: Any unit to be selected in Step 1.b has never been used in any of the previous tests.

Proof. The index of a unit to be selected in Step 1.b is x_k. Since the stack S is empty in this case, all the units that have been removed have participated in the tests which have failed. This means that all the simple tests carried out so far have failed. The number of these tests is F, which is now equal to k. Note that the units tested in the previous simple tests were selected either in Step 1.b or in the last part of Step 2 under the old values of k, which were at most k-1. Thus, if they were chosen in Step 1.b, then the F values were at most k-1. If chosen in the last part of Step 2, their subscripts were at most x_{k-1}. In both cases, the subscripts of these units were at most x_{k-1}. Q.E.D.

The above two lemmas imply that we can always perform a simple test in Step 1. We now prove that the algorithm always terminates. The following lemma is essential for this purpose.

Lemma 3: At the beginning of the (k+1)st simple test, $F+f^* \geq k$.

Proof. We first note that the parameter F indicates the number of simple tests which have failed. Furthermore, k indicates the number of simple tests that have been conducted so far. Therefore, the number of simple tests which have passed is k-F and those tests are contained in the compound tests whose corresponding binomial trees (actually, their roots) are stored in the stack S. If k = F, the stack is empty and here $f^* = 0$, which leads to the desired result. Thus, we consider the case when k > F. Note that f(u) denotes the number of units in a binomial tree rooted at u and their number is twice as large as the number of simple tests which are contained in its corresponding compound test. We also note that at the beginning of Step 1, the f(u) values stored in S are in increasing order from top to bottom. Therefore, the order of the binomial tree (or its corresponding compound test) stored at the bottom of S is $\lfloor \log(k-F) \rfloor + 1$. Therefore, $f^* = 2^{\lfloor \log(k-F) \rfloor + 1}$ and hence $f^* > 2^{\log(k-F)}$, or $f^* > k-F$, or $F+f^* > k$. Q.E.D.

Lemma 4: Algorithm 1 terminates before the (t+1)st simple test.

Proof. Prior to the (t+1)st simple test to be carried out in Step 1, we are either in Step 1.b or in the last part of Step 2. In the former case, $f^* = 0$ and hence, due to the proof of Lemma 3, $F = t$. This immediately implies that the algorithm has terminated in Step 1.b. In the latter case, by Lemma 3, $F + f^* > t$. This condition terminates the algorithm at the beginning of Step 2. Q.E.D.

The above lemma implies that k never exceeds t. Thus, the total number of simple tests never exceeds t. Therefore, Algorithm 1 is a different implementation of the Hakimi-Schmeichel algorithm. Using their theorem [22], we can obtain the following.

Lemma 5: Algorithm 1 uses $2t+1$ units and at most $2t - B_1(t)$ tests before it terminates.

Putting the above lemmas together, we now have the following theorem.

Theorem 1: Let S be a $D^*(n,t,X)$ system of n units, at most t of which are faulty, with $n \geq 2t+1$. Then Algorithm 1 always identifies a fault-free unit in S using $2t+1$ units and at most $2t - B_1(t)$ tests.

3. IDENTIFICATION OF ALL FAULTY UNITS

In the adaptive approach, one first identifies a fault-free unit, and then one uses this unit as a tester to identify the status of the remaining units. It is known [19,20] that the "indegree" of all vertices in G(U,T) must be $\geq t$. This is also a necessary condition for t-diagnosability in the "classical theory" [1,2]. In this section, we present a necessary and sufficient condition for identification of all faulty units after locating a fault-free unit. This is the sufficient condition for t-diagnosability in the classical theory which was obtained by Hakimi and Amin [2].

A sequence of vertices and edges in G(U,T), $u_{i_1}, (u_{i_1},u_{i_2}), u_{i_2}, \ldots, u_{i_{k-1}}, (u_{i_{k-1}},u_{i_k}), u_{i_k}$ is called a dipath from u_{i_1} to u_{i_k}. If for any pair of vertices u_i and u_j there is a dipath from u_i to u_j in G, G is called strongly connected. Let G-X, $X \subset U$, denote the digraph obtained from G by removing the set X and all the edges with at least one vertex in X. The connectivity c(G) of a digraph G is the minimum number of vertices whose removal from G produces a digraph which is not strongly connected. It may be noted that if $c(G)=t$, then the indegree (as well as the outdegree) of any vertex of G must be $\geq t$.

Theorem 2: Let S be a system of n units, at most t of which are faulty, with $n \geq 2t+1$. Given a fault-free unit, all faulty units in S can be identified if and only if $c(G) \geq t$.

Proof. Suppose that all faulty units in S can be identified after locating a fault-free unit. Assume, however, that $c(G) < t$. Then, in G there is a pair of vertices u and v and a set X of vertices with $|X| < t$ such that there is no dipath from u to v in G-X. If u is the given fault-free unit and the units in X are all faulty, then it is impossible to identify the status of unit v, which is a contradiction to the assumption. Thus, $c(G) \geq t$. Now suppose $c(G) \geq t$. Let u be the given fault-free unit. Then,

$c(G) \geq t$ implies that for any other unit v in G, there are at least vertex-disjoint dipaths from u to v. Therefore, along each of these dipaths we can perform a sequence of tests as long as the result of each test is fault-free. Then, either a test is carried out by some fault-free unit on unit v, or no such test is performed. In the former case, the status of unit v is identified. The latter case occurs only when there are exactly t vertex-disjoint dipaths from u to v and a series of tests along each such dipath are terminated with the result of the last test being faulty. In such a case, all the t faulty units have been identified and hence unit v is fault-free. Q.E.D.

4. ADAPTIVE DIAGNOSIS FOR UNDIRECTED GRAPH MODELS

Consider a system of n identical units in which tests are conducted for pairs of units. The same sequence of test signals are applied to a pair of units u and v and their responses are compared; the test result, denoted by R(u,v), is 0 if they agree in the responses, and 1 if they disagree. The comparison assignment for possible tests is represented by an undirected graph where the vertices correspond to the units and an undirected edge between two vertices signifies that the same test signals may be applied to the corresponding units. The test result, R(u,v), for a pair of units u and v is expressed as the weight of edge (u,v) in the graph. Assuming that the nature of faults is permanent, we make three different assumptions on the result of a test. Let g and b denote a fault-free and a faulty units, respectively.

Assumption 1	Assumption 2	Assumption 3
R(g,g) = 0	R(g,g) = 0	R(g,g) = 0
R(g,b) = 0 or 1	R(g,b) = 1	R(g,b) = 1
R(b,b) = 0 or 1	R(b,b) = 0 or 1	R(b,b) = 1

Assuming that there are no more than t faulty units in a system of n units, one adaptively chooses pairs of units for testing and seeks the results of these tests until one identifies a fault-free unit. This unit is then used as a tester to identify all faulty units in the system. Nakajima [25] considered this adaptive fault identification problem for the undirected graph model under the above three assumptions. We briefly discuss some of his results.

The undirected graph model under Assumption 1 was studied by Chwa and Hakimi [16] in the application of the theory of t-diagnosable systems to fault-tolerant computing. They obtained a characterization theorem in this model for t-diagnosability by modifying the Hakimi-Amin theorem [2] in the PMC model. We can obtain such a characterization theorem in the undirected graph model under Assumption 3 by modifying the Barsi-Grandoni-Maestrini theorem in the BGM model [5]. Therefore, all the existing adaptive procedures for the PMC model are applicable to the undirected graph model under Assumption 1, while those for the BMG model are applicable to the same model under Assumption 3. Some interesting results are also obtained for the undirected graph model under Assumption 2.

5. CONCLUSIONS

We have considered the problem of adaptively identifying all faulty units in a system of n units and nt test links. We have shown that a variation of the Hakimi-Schmeichel algorithm correctly identifies a fault-free unit in such a system of class $D^*(n,t,X)$. One of our immediate goals is to prove that a similar algorithm adaptively identifies a fault-free unit in a general $D(n,t,X)$ system.

We have presented a characterization theroem for identification of all faulty units after locating a fault-free unit. Recently, the Hakimi-Nakajima and Hakimi-Schmeichel algorithms were shown to identify all faulty units in a $D_{\delta t}$ system [20,23]. We should note that a $D_{\delta t}$ system satisfies the condition of our theorem. Another goal of ours is to prove that a $D^*(n,t,X)$ system and even a $D(n,t,X)$ system satisfy such a condition.

Finally, we have described the adaptive fault identification problem for the undirected graph models. We have briefly summarized some of Nakajima's [25] results on this problem. In this way, we have shown that the adaptive approach is applicable to the different models. Therefore, the applicability of this approach to the various other models should be investigated.

REFERENCES

[1] F. P. Preparata, G. Metze and R. T. Chien, "On the connection assignment problem of diagnosable systems," IEEE Trans. Electron. Comput., Vol. EC-16, pp. 848-854, December, 1967.

[2] S. L. Hakimi and A. T. Amin, "Characterization of connection assignment of diagnosable systems," IEEE Trans. Comput., Vol. C-23, pp. 86-88, January, 1974.

[3] J. D. Russel and C. R. Kime, "System fault diagnosis: Closure and diagnosability with repair," IEEE Trans. Comput., Vol. C-24, pp. 1078-1089, November, 1975.

[4] J. D. Russel and C. R. Kime, "System fault diagnosis: Masking, exposure and diagnosability without repair," IEEE Trans. Comput., Vol. C-24, pp. 1155-1161, December, 1975.

[5] F. Barsi, F. Grandoni and P. Maestrini, "A theory of diagnosability of digital systems," IEEE Trans. Comput., Vol. C-25, pp. 585-593, June, 1976.

[6] S. N. Maheshwari and S. L. Hakimi, "On models for diagnosable systems and probabilistic fault diagnosis," IEEE Trans. Comput., Vol. C-25, pp. 228-236, March, 1976.

[7] U. Manber, "System diagnosis with repair," IEEE Trans. Comput., Vol. C-29, pp. 934-937, October, 1980.

[8] P. Ciompi and L. Simoncini, "Analysis and optimal design of self-diagnosable systems with repair," IEEE Trans. Comput., Vol. C-28, pp. 362-365, May 1979.

[9] H. Fujiwara and K. Kinoshita, "Connection assignment for probabilistically diagnosable systems," IEEE Trans. Comput., Vol. C-27, pp. 280-283, March, 1978.

[10] H. Fujiwara and K. Kinoshita, "Some existence theorems for probabilistically diagnosable systems," IEEE Trans. Comput., Vol. C-27, pp.379-384, April, 1978.

[11] J. E. Smith, "Universal system diagnosis algorithms," IEEE Trans. Comput., Vol. C-28, pp. 374-378, May, 1979.

[12] C. S. Holt and J. E. Smith, "Diagnosis of systems with asymmetric invalidation," IEEE Trans. Comput., Vol. C-30, pp. 679-690, September, 1981.

[13] S. Mallela and G. M. Masson, "Diagnosable systems for intermittent faults," IEEE Trans. Comput., Vol. C-27, pp. 560-566, June, 1978.

[14] K.-Y. Chwa and S. L. Hakimi, "On fault identification in diagnosable systems," IEEE Trans. Comput., Vol. C-30, pp. 414-422, June 1981.

[15] J. T. Butler, "Speed-efficiency-complexity tradeoffs in universal diagnosis algorithms," IEEE Trans. Comput., Vol. C-30, pp. 590-596, August, 1981.

[16] K.-Y. Chwa and S. L. Hakimi, "Schemes for fault-tolerant computing: A comparison of modularly redundant and t-diagnosable systems," Inform. Contr., Vol. 49, pp. 212-238, June 1981.

[17] A. T. Dahbura and G. M. Masson, "Greedy diagnosis as the basis of an intermittent-fault/transient-upset tolerant system design," IEEE Trans. Comput., to appear.

[18] K. Nakajima, "A new approach to system diagnosis," Proc. 19th Annual Allerton Conf. Commun., Contr. and Comput., September, 1981, pp. 697-706.

[19] S. L. Hakimi and K. Nakajima, "Adaptive diagnosis: A new theory of t-fault diagnosable systems," Proc. 20th Annual Allerton Conf. Commun., Contr., and Comput., October, 1982, pp. 231-240.

[20] S. L. Hakimi and K. Nakajima, "On adaptive system diagnosis," IEEE Trans. Comput., to appear.

[21] P. M. Blecher, "On a logical problem," Discrete Math., Vol. 43, pp. 107-110, 1983.

[22] S. L. Hakimi and E. F. Schmeichel, "An adaptive algorithm for system level diagnosis," Jour. Algorithms, to appear.

[23] K. Nakajima and V. P. Krothapalli, "On adaptive fault identification for certain classes of diagnosable systems," Proc. 17th Annual Conf. Inform. Sci. and Syst., March, 1983, p. 455.

[24] J. Vuillemin, "A data structure for manipulating priority quenes," Comm. of the ACM, Vol. 21, pp. 309-315, April, 1978.

[25] K. Nakajima, "Adaptive system diagnosis for comparison assignment models," unpublished notes, Texas Tech University, August 1983.

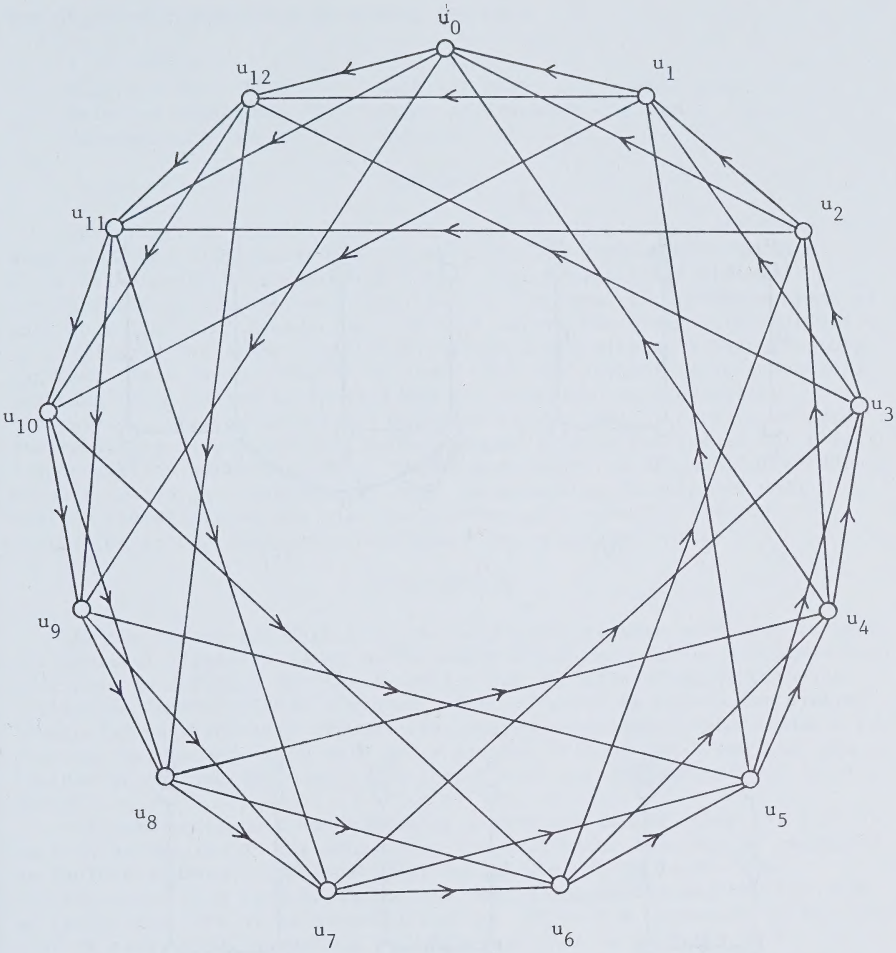

u_0
u_1
u_2
u_3
u_4
u_5
u_6
u_7
u_8
u_9
u_{10}
u_{11}
u_{12}

Fig. 1. D*(13,3, {1,2,4}) system.

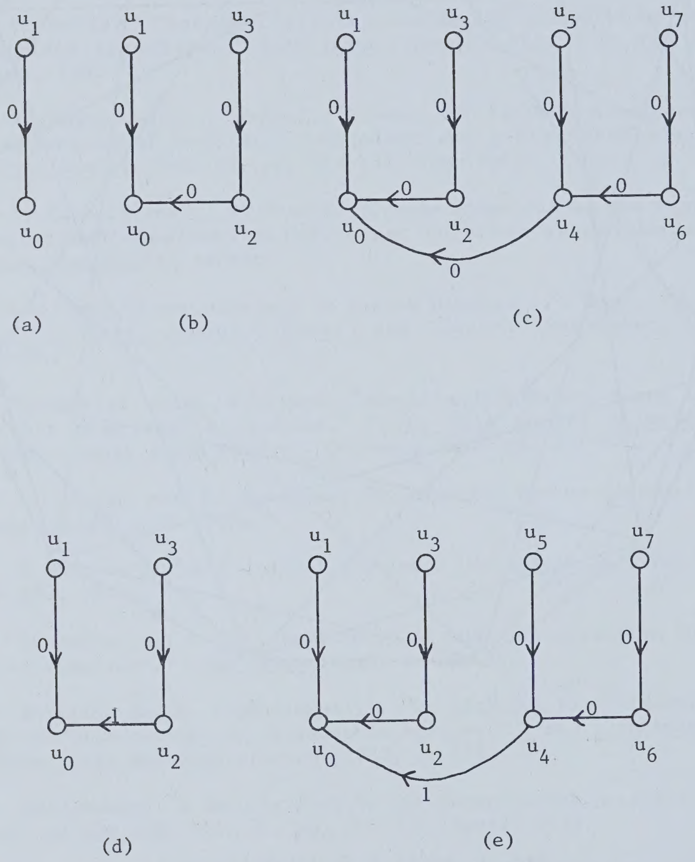

u_0
u_1
u_2
u_3
u_4
u_5
u_6
u_7
0
1
(a)
(b)
(c)
(d)
(e)

Fig. 2. Examples of binomial trees to be generated during the execution of Algorithm 1.

THE DIAGNOSTIC CONFIDENCE OF T-FDOR SYSTEMS*

I. S. HWANG
Dept. of Elec. Engr.
Univ. of Minnesota
Minneapolis, MN

C. R. KIME
Dept. of Elec. and Comput. Engr.
Univ. of Wisconsin-Madison
Madison, WI

ABSTRACT

In situations where tests are incomplete, diagnostic confidence, a measure of the likelihood that a system will be diagnosable, is an alternative measure to diagnosability. Such a measure is also of considerable importance because it is one component of system coverage as used in reliability evaluation. In this paper, the diagnostic confidence is considered for a class of t-fault diagnosable without repair (T-FDOR) systems. Test incompleteness is dealt with via probabilistic components and test invalidation is handled via deterministic diagnosability information, whereby the use of unavailable statistical data is avoided. The resulting diagnostic confidence measure is thus confidence for t-fault diagnosability without repair. Determination of the diagnostic confidence measure, which is comparatively easy, is presented in terms of unit failure probabilities and test fault coverages. General results pertaining to the diagnostic confidence are also discussed.

INTRODUCTION

As the demand for highly available digital systems continues to grow, system-level diagnosis is an increasingly important problem. System-level diagnosis is intended for the detection and location of malfunctioning subsystems (units) within a system. Most of previous models for system-level diagnosis consider systems that can be partitioned into a number of disjoint units. For each unit it is assumed that at least one test exists capable of testing that unit, and these tests are implemented via other units.

If some units used in performing a test are faulty, then the test is said to be invalid in the sense that the unit under test may be evaluated as fault-free when it is faulty, or faulty when it is fault-free. Further, even if a test is valid, the unit being tested may be evaluated as fault-free when it is actually faulty. If such a situation arises, the test is said to be incomplete. The factors that cause tests to be incomplete include: some classes of faults such as a class of intermittent faults are inherently difficult to detect, tests are usually derived under the assumption that certain faults are likely to occur, test generation even under this assumption is generally complex and generating tests with full fault coverages is extremely difficult, and too much time is often required to exercise tests sufficiently. The test invalidation and incompleteness greatly increase the difficulty of system-level diagnosis.

Several studies in system-level diagnosis, notably [1-4,9], address the analysis and design of diagnosable systems and the establishment of diagnosis strategies and algorithms. These studies assume explicitly test completeness and deal primarily with test invalidation. As test incompleteness situations are very likely, or perhaps inevitable, in real applications, the treatment of incomplete diagnostic tests is essential for system-level diagnosis theory to be more useful and applicable in practice. For instance, a system which is diagnosable to some degree

---

* This work was supported in part by NSF Grant MCS 79-09902.

under the assumption that each test is complete may not be diagnosable at all if one of the tests is indeed incomplete. Also, in such a situation, a diagnosis algorithm developed assuming test completeness may collapse when unexpected test outcomes are encountered.

Consideration of incomplete tests is given in [5-8]. In [5,6], both test incompleteness and invalidation are handled via a probabilistic approach, and probabilistic determination of a diagnosis algorithm and derivation of probabilistic diagnosability measures are discussed for diagnosis without repair case. It seems, however, that the statistical data required to deal with test invalidation is voluminous and difficult to obtain, and determination of the algorithm and measures involves computation of extreme complexity. Also, the s-dependence among test results is not appropriately taken into account, although both test incompleteness and invalidation are assumed to arise in the presence of a class of permanent faults, and thus test results are in general strongly correlated.

In [7,8], test incompleteness is assumed to come from the ephemeral nature of intermittent faults, and the test incompleteness as well as invalidation is dealt with in a deterministic manner as in [1-3]. The diagnosis strategy proposed is based on repeated applications of diagnostic routines until test results which a permanent fault situation can produce are obtained. The strategy avoids incorrect diagnosis of any fault-free unit as faulty, while allowing incomplete diagnosis, that is, some faulty units may remain undetected at the completion of diagnosis. In view of recent work [10,11], the suggestion of repeated applications of tests seems somewhat unacceptable in practice. Also, diagnosable systems in the above sense generally require more tests and more stringent test connection.

In this paper, an approach is presented which uses probabilistic modeling for test incompleteness and deterministic modeling for test invalidation, thus exploiting the best features of both modeling techniques. The source of test incompleteness is to be broadly interpreted as previously discussed. After the general model proposed in [3] is extended to handle incomplete tests, diagnosis without repair in incomplete test situations is discussed and a general diagnosis algorithm is developed that can be applicable to a broad class of systems. The diagnostic confidence of T-FDOR systems is then defined and its determination in terms of unit failure probabilities and test fault coverages is presented.

DIAGNOSTIC MODEL

The model to be used is the one that is extended from the Russell-Kime model [3] with additional model components and some different interpretations. The model definition that is necessary in this paper is briefly described.

A system is composed of a set of n potentially faulty units $f_1, f_2, \ldots, f_n$ and has a set of m pass-fail tests $t_1, t_2, \ldots, t_m$. A broad interpretation is given to both units and tests. For example, a unit may be a functional unit such as an arithmetic logic unit, microprocessor, or a computer, and a test may be a diagnostic program, a comparison test, or the use of checking circuits and error-detection codes. It is assumed that unit failures are s-independent of one another, and the failure probability of unit f_i is denoted by $P(f_i)$. Each unit has at least one test on it and each test is applied to only one unit. Also, each test is performed by a set of other units and thus no unit tests itself. This relationship between units and tests may be represented by a bipartite graph [9] or binary arrays as in [3]. Fig. 1 and Fig. 2 show the bipartite graph representation of two systems to be used as examples.

A test is said to be valid if each unit used in performing the test is fault-free, and invalid otherwise. If a test is invalid, the test may fail when the assumed unit is fault-free and pass when the assumed unit is faulty. A test is said to be complete if the test definitely detects the assumed faulty unit when it is valid, and incomplete otherwise. If a test is incomplete, then the test may pass when the assumed unit alone is faulty. It is assumed that the results of invalid or incomplete tests are not deterministically correlated with one another, thus the result of one test does not imply the result of another test. The conditional probability that test t_j will fail given the assumed unit alone is faulty is called the fault coverage of t_j, and is denoted by $P(t_j)$. Thus, the fault coverage of a test is a measure of the capability of the test to detect the assumed faulty unit when it is valid.

Multiple units may be faulty in a system. A fault pattern F^i is a set of units that can be simultaneously faulty at any particular time, and in particular, F^0 is used to denote the empty fault pattern. For any integer t, $0 \leq t \leq n$, F(t) denotes the collection of all fault patterns that contain t or fewer units. A test pattern T^j is a set of tests that can fail simultaneously in the presence of a fault pattern, and thus is a test set result. T^0 is used to denote the empty test pattern and T(t) denotes the collection of all test patterns that the fault patterns in F(t) can produce.

An array representation of the relationship between the fault patterns in F(t) and the test patterns in T(t) is provided by the G(t)-array having as its (i,j)-entry G_j^i = 0, 1, or X if, for fault pattern F^i in F(t) present, test t_j definitely passes, definitely fails, or possibly either passes or fails, respectively. For the present application: $G_j^i = 0$ if t_j is valid in the presence of F^i and the assumed unit is not in F^i, $G_j^i = 1$ if t_j is valid in the presence of F^i and complete for the assumed unit in F^i, and $G_j^i = X$ if t_j is invalid in the presence of F^i or incomplete for the assumed unit in F^i. Note that a row G^i containing k entries of X represents 2^k test patterns which are all possible with F^i present in the system, in other words, G^i represents the collection of all test patterns that can occur in the presence of F^i. In this paper, $T(t)_c$ and $G(t)_c$-array are used to denote the T(t) and G(t)-array with each test assumed to be complete, respectively.

DIAGNOSIS WITHOUT REPAIR

One of the diagnosis methods that are commonly studied is diagnosis without repair [2]. The diagnosis method involves one application of the test set followed by an attempt to identify all faulty units based on the test results without falsely identifying any fault-free unit as faulty. Difficulties arise in doing this, especially in incomplete test situations, because a great variety of test patterns can potentially occur in the presence of a fault pattern and a number of fault patterns can produce the same test patterns. For example, consider the system of Fig. 2 having 5 units and 10 tests. The number of test patterns that can occur when 4 or 5 units are faulty is 1024. If one of the tests fails and the others pass, then any of 24 fault patterns may be present in the system.

Accordingly, a common practice to alleviate the extent of these difficulties is to restrict the fault patterns and test patterns to be considered to those in $F(t)$ and those in $T(t)_c$, respectively, for some t such that each test pattern in $T(t)_c$ can occur in the presence of only one fault pattern in $F(t)$. This requires that a system be T-FDOR. A system is T-FDOR if all faulty units are detected and precisely identified by one application of the test set provided t or fewer units are faulty. Note that the definition assumes that each test is complete. The systems of Fig. 1 and Fig. 2 are, for instance, 1-FDOR and 2-FDOR, respectively.

Now, the diagnosis strategy for T-FDOR systems is described as: 1) apply the test set and 2) if the resulting test pattern is in $T(t)_c$, then identify all faulty units at once without any false identification, and otherwise, identify no faulty unit.

Of importance are algorithms which implement the above diagnosis strategy. A diagnosis algorithm of this class is presented which has as its inputs $F(t)$, the $G(t)_c$-array, and a given test pattern T^j and as its output a set F^p of identified faulty units. In the algorithm, T^j is assumed to be represented by a m-dimensional vector such that its k-th element $T^j_k = 1$ if test t_k fails and $T^j_k = 0$ otherwise, and n' is the number of fault patterns in $F(t)$. The cubical intersection $\sqcap$ is defined in [13].

```
PROCEDURE ALGO-OR($F(t)$,$G(t)_c$-array,$T^j$,$F^p$)
    $G^{n'} \leftarrow T^j$;
    $i \leftarrow 0$;
    WHILE $G^i \sqcap T^j = \emptyset$ DO
        $i \leftarrow i + 1$;
    END
    IF $i = n'$ THEN $F^p \leftarrow \emptyset$
              ELSE $F^p \leftarrow F^i$
    END
END ALGO-OR
```

The above algorithm basically uses a sequential search method to find G^i such that $G^i \sqcap T^j \neq \emptyset$. Its worst-case computing time is $O(n'm)$. The addition of the dummy row $G^{n'}$ eleminates the need for testing the end of the $G(t)_c$-array in the WHILE loop and thus simplifies the search. It also guarantees the algorithm termination even if T^j is not in $T(t)_c$. The algorithm is computationally feasible in a low/medium test invalidation environment which is very likely in reality in the light of extremely low unit failure probabilities, and is applicable to a broad class of T-FDOR systems as opposed to other existing algorithms of the same class.

It should be emphasized, however, that the diagnosis strategy for T-FDOR systems does not always result in a fault-free T-FDOR system even if the system contains t or fewer faulty units. This is because some test pattern not in $T(t)_c$ can indeed occur in the presence of a fault pattern in F(t) and consequently, no faulty unit can be identified.

DIAGNOSTIC CONFIDENCE OF T-FDOR SYSTEMS

In this section, the confidence of diagnosis of T-FDOR systems is considered. The diagnosis strategy discussed in the previous section is assumed to be used in the actual diagnosis.

Formally, the diagnostic confidence of a T-FDOR system (DC-OR) is the conditional probability that all faulty units will be detected and precisely identified by one application of the test set given t or fewer units are faulty. The DC-OR is an alternative measure to diagnosability without repair that is useful in situations where some or all tests are incomplete. It is also of considerable importance because it is one component of system coverage. Obviously, the DC-OR would be 1.0 if each test were complete. In the following, the DC-OR is determined in terms of unit failure probabilities and test fault coverages and its general results are discussed.

Let A^i be the event that fault pattern F^i is present in the system and B^i the event that F^i is identified by one application of the test set. Then, the foregoing definition of DC-OR can be recast in a more workable form as follows.

$$\text{DC-OR} = P(\bigcup (A^i \cap B^i) \text{ over all } F^i \text{ in } F(n) \mid \bigcup A^i \text{ over all } F^i \text{ in } F(t)) \quad (4.1)$$

By definition of conditional probability and since A^is are mutually exclusive events,

$$\text{DC-OR} = \frac{\sum P(A^i)P(B^i \mid A^i) \text{ over all } F^i \text{ in } F(t)}{\sum P(A^i) \text{ over all } F^i \text{ in } F(t)} \quad (4.2)$$

In the above expression, $P(A^i)$ is easily obtained from $P(f_j)$'s by the assumption of s-independence of unit failures. Thus, of concern is the determination of $P(B^i \mid A^i)$. By definition, T^0 is one and only one test pattern that can occur for fault pattern F^0 present and F^0 is identified upon the occurrence of T^0. Thus,

$$P(B^i \mid A^i) = 1.0 \text{ for } F^i = F^0 \quad (4.3)$$

The following theorem is necessary for the subsequent development and shows the existence of some valid tests on each of t or fewer faulty units that a t-FDOR system may contain.

Theorem 1: If fault pattern F^i in F(t) is present in a T-FDOR system, then each faulty unit has at least one valid test on it in the presence of F^i.

Two theorems follow again that enable the determination of $P(B^i \mid A^i)$ for $F^i \neq F^0$ in F(t) in terms of more readily available conditional probabilities.

Theorem 2: Given fault pattern $F^i \neq F^0$ in F(t) is present in a T-FDOR system, F^i is identified by one application of the test set if and only if all valid tests on faulty units in the presence of F^i fail.

Let C^i be the event that all valid tests on faulty units in the presence of F^i fail. Theorem 2 indicates that given A^i for $F^i \neq F^0$ in F(t), B^i and C^i are the same events, which in turn gives

$$P(B^i \mid A^i) = P(C^i \mid A^i) \text{ for } F^i \neq F^0 \text{ in } F(t) \tag{4.4}$$

Theorem 3: If r sets of tests are valid on r faulty units $f_k, \ldots, f_{k'}$, respectively, in the presence of a fault pattern, then the r sets of test results are conditionally s-independent of one another given the fault pattern is present.

An additional definition is made. The joint fault coverage of two or more tests on a unit is the conditional probability that the tests will fail given the unit alone is faulty. Note the tests are valid in that situation. Let $P(c_j \mid F^i)$ denote the joint fault coverage of all valid tests on faulty unit f_j in the presence of $F^i \neq F^0$. The conditional s-independence asserted by Theorem 3 makes possible the determination of $P(C^i \mid A^i)$ for $F^i \neq F^0$ in terms of $P(c_j \mid F^i)$'s. That is,

$$P(C^i \mid A^i) = \prod P(c_j \mid F^i) \text{ over all } f_j \text{ in } F^i \text{ for } F^i \neq F^0 \text{ in } F(t) \tag{4.5}$$

By substituting (4.3) directly and (4.5) through (4.4) into (4.2) for the respective cases of $P(B^i \mid A^i)$, the DC-OR is obtained as follows.

$$\text{DC-OR} = \frac{P(F^0) + \sum P(F^i)\left(\prod P(c_j \mid F^i) \text{ over all } f_j \text{ in } F^i\right) \text{ over all } F^i \neq F^0 \text{ in } F(t)}{\sum P(F^i) \text{ over all } F^i \text{ in } F(t)} \tag{4.6}$$

where $P(F^i)$ replaces $P(A^i)$ in (4.2).

In general, the results of valid tests on faulty unit f_j in the presence of $F^i \neq F^0$ are conditionally s-dependent on one another, and the exact value of $P(c_j \mid F^i)$ may not be obtained from $P(t_k)$'s, where t_ks are the valid tests on f_j in the presence of F^i, unless the correlations among the test results are known. In such a situation, the following theorem is useful in obtaining the lower and upper bounds on $P(c_j \mid F^i)$ from $P(t_k)$'s.

Theorem 4: Let $t_k, \ldots, t_{k'}$ be r valid tests on faulty unit f_j in the presence of fault pattern F^i. Then,

$$\operatorname{Max}(0,\ P(t_k)+\ldots+P(t_{k'})-r+1) \leq P(c_j \mid F^i) \leq \operatorname{Min}(P(t_k), \ldots, P(t_{k'})) \tag{4.7}$$

The lower and upper bounds on $P(c_j \mid F^i)$ obtained using Theorem 4 are general in the sense that no specific assumption is made in their derivation. On the other hand, the lower bound may be too conservative since $P(t_k)$'s are in general positively correlated.

Now, the procedure for determining the DC-OR from the unit failure probabilities and test fault coverages is as follows: 1) determine t, the diagnosability without repair, of a given system, 2) find, possibly on the basis of the $G(t)_c$-array, which tests are valid on each faulty unit in the presence of each fault pattern in F(t), and 3) compute the DC-OR using (4.6) and (4.7).

Although the $G(n)_c$-array can be used to determine t, its direct use may be cumbersome. More readily applied results are available in the literature [2]. Table 1 shows the numerical values of DC-OR of the two systems of Fig. 1 and Fig. 2 along with the unit failure probabilities and test fault coverages used in the computation.

The computational complexity of using (4.6) and (4.7) to determine the DC-OR is $O(n'm't)$, where n' is the number of fault patterns in F(t) and m' is the maximum number of tests on any unit. It seems that any method to evaluate the DC-OR is of at least this complexity. Indeed, the approach proposed in [5,6] to evaluate the probabilistic diagnosability measures similar but not identical to the DC-OR considers all test patterns in T(n), and its computational complexity is $O(n''m''m)$, where n'' is the number of fault pattern in F(n) and m'' is the number of test patterns in T(n). More critically, the approach requires far more unavailable statistical data such as conditional probabilities of invalid test results.

General results pertaining to the DC-OR indicate that as the numbers of incomplete tests on units increase, the sets of tests should be carefully derived so that the fault coverages of individual tests of each set are high and positively correlated. Otherwise, the DC-OR is significantly degraded. In other words, higher diagnosability without repair should not be ascertained simply by adding more tests and assuming these tests are complete. Also, excess tests that are not needed to achieve the diagnosability level required can potentially degrade the DC-OR because of the exact match requirement of valid test results, and thus such tests should be deleted. These observations are considered to be quite significant in the design of T-FDOR systems since such a system tends to require a considerable number of tests on each unit.

In many usual cases, the unit failure probabilities are so low that the probability of no unit being faulty is much higher than that of at least one unit being faulty, that is, a system is much more likely to be fault-free than faulty. In such situations, the DC-OR may not significantly indicate the merit of diagnosis, and thus may be misleading. Therefore, the normaized version of DC-OR may be defined and used, as done in [12] for diagnosis with repair case, by assuming that a system is initially faulty. The expression of normalized DC-OR is easily obtained by deleting the term $P(F^0)$ from both numerator and denominator of (4.6). The numerical values of normalized DC-OR of the two example systems are also shown in Table 1.

CONCLUSIONS

A computaionally efficient approach has been given to evaluate the diagnostic confidence of T-FDOR systems in terms of only the statistical data that is usually assumed to be available. In general, it is extremely difficult and voluminous to obtain and handle the conditional probabilities of invalid test results and their correlations, and thus their use has been avoided. Also, it has been indicated that multiple tests on a unit should be carefully derived and that excess tests which are not needed to achieve the required diagnosability should be deleted. Otherwise, the joint fault coverage of tests can be poor even though the fault coverages of individual tests are very high. Poor joint fault coverages can, in turn, result in low diagnostic confidence.

REFERENCES

[1] F. P. Preparata, G. Metze, and R. T. Chien, "On the connection assignment problem of diagnosable systems," IEEE Trans. Electronic Comput., Vol. EC-16, pp. 848-854, December, 1967.

[2] J. D. Russell and C. R. Kime, "System fault diagnosis: masking, exposure, and diagnosability without repair," IEEE Trans. Comput., Vol. C-24, pp. 1155-1161, December, 1975.

[3] J. D. Russell and C. R. Kime, "System fault diagnosis: closure and diagnosability with repair," IEEE Trans. Comput., Vol. C-24, pp. 1078 -1089, November, 1975.

[4] F. Barsi, F. Grandoni, and P. Maestrini, "A theory of diagnosability of digital systems," IEEE Trans. Comput., Vol. C-25, pp. 585-593, June, 1976.

[5] M. L. Blount, "Probabilistic treatment of diagnosis in digital systems," Proc. 7th Int. Cof. on Fault-Tolerant Computing, pp. 72-77, June, 1977.

[6] M. L. Blount, "Modeling of diagnosis in fail-softly computer systems," Proc. 8th Int. Cof. on Fault-Tolerant Computing, pp. 53-58, June, 1978.

[7] S. Mallela and G. M. Masson, "Diagnosable systems for intermittent faults," IEEE Trans. Comput., Vol. C-27, pp. 560-566, June, 1978.

[8] S. Mallela and G. M. Masson, "Diagnosis without repair for hybrid fault situations," IEEE Trans. Comput., Vol. C-29, pp. 461-470, June, 1980.

[9] C. S. Holt and J. E. Smith, "Diagnosis of systems with asymmetric invalidation," IEEE Trans. Comput., Vol. C-30, pp. 675-690, September, 1981.

[10] B. C. Bossen and M. Y. Hsiao, "Model for transient and permanent error-detection and fault-isolation coverage," IBM J. Res. Develop., Vol. 26, pp. 67-77, January, 1982.

[11] N. N. Tendolkar and R. L. Swann, "Automated diagnostic methodology for the IBM 3081 Processor Complex," IBM J. Res. Develop., Vol. 26, pp. 78-88, January, 1982.

[12] I. S. Hwang and C. R. Kime, "On a diagnostic confidence measure for a class of digital systems," Proc. 20th Ann. Conf. on Comm., Contr., and Comput., pp. 241-250, October, 1982.

[13] D. L. Dietmeyer, Logic Design of Digital Systems, 2nd Ed., Boston: Allyn and Bacon, 1971.

APPENDIX

Proof of Theorem 1:

Since test completeness does not affect test validity, each test is assumed to be complete without loss of generality. The contrapositive is proven. Assume that F^i in F(t) is present in a T-FDOR system. Then, by the definition of a T-FDOR system, each test pattern in $T(t)_c$ that can occur in the presence of F^i cannot occur in the presence of any other fault pattern in F(t). Now, suppose that some faulty unit f_k in F^i has all tests on it invalid in the presence of F^i and consider another fault pattern $F^{i'}$ such that $F^{i'} = F^i - \{f_k\}$. Since F^i is in F(t), $F^{i'}$ is also

in F(t). Let $t^+(f_k)$ denote the set of tests which are applied to f_k and $t^-(f_k)$ the set of tests which are performed at least in part by f_k. Then, since all tests in $t^+(f_k)$ or in $t^-(f_k)$ are invalid in the presence of F^i, $G^i_j = X$ and thus $G^i_j \sqcap G^{i'}_j \neq \emptyset$ for all t_j such that t_j is in $t^+(f_k)$ or $t^-(f_k)$. Further, $G^i_j = G^{i'}_j$ and thus $G^i_j \sqcap G^{i'}_j \neq \emptyset$ for all t_j such that t_j is not in $t^+(f_k)$ nor in $t^-(f_k)$. Therefore, $G^i_j \sqcap G^{i'}_j \neq \emptyset$ for all t_j in the test set and thus $G^i \sqcap G^{i'} \neq \emptyset$. This implies that there exists some test pattern in $T(t)_c$ that can occur in the presence of F^i and $F^{i'}$, which contradicts the assumption that each test pattern in $T(t)_c$ that can occur in the presence of F^i cannot occur in the presence of any other fault pattern in F(t).
Q.E.D.

Proof of Theorem 2:

For sufficiency, assume that all valid tests on faulty units in the presence of F^i fail. Then, any test pattern that can occur is one of the test patterns in $T(t)_c$. Therefore, F^i is identified by one application of the test set. For necessity, the contrapositive is proven. Suppose that some valid test on a faulty unit in the presence of F^i passes. Then, any test pattern that can occur is not in the set of test patterns in $T(t)_c$ that F^i can produce. Thus, F^i cannot be identified by one application of the test set.
Q.E.D.

Proof of Theorem 3:

Note that r sets of tests are disjoint by definition. Assume that r sets of tests are valid on r faulty units, respectively, in the presence of a fault pattern. Then, each set of test results is determined soly by each respective fault unit and does not depend on the other faulty units. Therefore, the sets of test results are conditionally s-independent of one another give the fault pattern is present.
Q.E.D.

Proof of Theorem 4:

The proof is similar to that in [12].
Q.E.D.

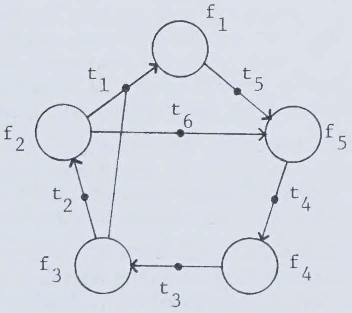


Fig. 1

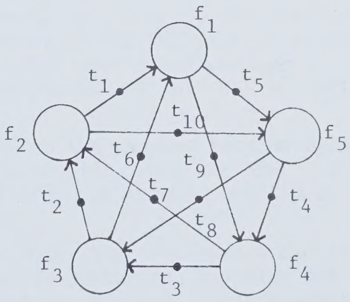


Fig. 2

The system of Fig. 1

Unit failure probabilities used:
f_1: 0.010 f_2, f_3: 0.005 f_4, f_5: 0.008

Test fault coverages used:
t_1: 0.920 t_2, t_3, t_4: 0.900 t_5, t_6: 0.850

$0.995 \leq \text{DC-OR} \leq 0.996$
$0.861 \leq \text{NC-OR}^* \leq 0.894$

The system of Fig. 2

Unit failure probabilities used:
f_1, f_2, f_3, f_4, f_5: 0.010

Test fault coverages used:
t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}: 0.900

$0.990 \leq \text{DC-OR} \leq 0.995$
$0.798 \leq \text{NC-OR}^* \leq 0.898$

* NC-OR denotes the normalized DC-OR.

Table 1 Numerical values of DC-OR and NC-OR of the two systems of Fig. 1 and Fig. 2.

RECENT MODIFICATIONS OF THE L-A-S (LINEAR ALGEBRA AND SYSTEMS) LANGUAGE AND ITS USE IN CAD OF CONTROL SYSTEMS

S.P. BINGULAC
University of Belgrade,
Department of Organizational Sciences,
Beograd, Jove Ilica 154, Yugoslavia

ABSTRACT

The paper reviews basic features of a high level interactive conversational language L-A-S (Linear Algebra and Systems) used in analysis and design of linear control systems. The L-A-S software is written in Fortran but its use does not require knowledge of standard programming languages. Modular structure of the language permits easy modification, updating and extentions. The syntax and semantics are simple and straightforward so that familiarity with the language may be easily attained. The language is implementable on both large scale and mini computer systems. Particular emphasis is given to recently introduced modifications and extentions which have resulted from suggestions of various users. These extentions improve the language flexibility and facilitate its application in solving various control problems. Illustrative examples of L-A-S use are included.

1. INTRODUCTION

L-A-S (Linear Algebra and Systems) is an interactive conversational software used in computer -aided design of control systems. It provides an easy access to any available Fortran subroutine performing a specific task. The present version of L-A-S uses subroutines from EISPACK and LINPACK libraries as well as more sophisticated subroutines developed using these two libraries. The standard L-A-S capabilities encompass standard matrix operations, solution of simultaneous linear algebraic equations and eigenvalue problems, calculation and plotting of dynamic system responses, calculation of rank, range and null spaces, singular value decomposition, solution of Riccati and Lyapunov matrix equation, similarity transformation in the state space, algorithms for pole placement using either state or output feedback, determination of minimal realization, calculation of matrix transfer function, transformation of continuous into discrete realization and vice versa, as well as others. The detailed list of all operators, referred to as L-A-S operators, is given in /1/ and /2/. During the L-A-S software development particular attention was given to simplify the procedure of adding new operators to the software. As a result, any available Fortran subroutine may be easily incorporated in the software as an additional operator. In fact, most of available L-A-S operators have been included in the software according to suggestions and interest of various users the author has contacted so far.

A recent software extention permits the user to combine existing L-A-S operators into L-A-S user defined suborutine. From the user's point of view, once created, the user defined subroutine is used exactly in the same manner as elementary L-A-S operator.

One of the L-A-S's primary uses is in the development and testing of control analysis and design algorithms. It usefulness has also been verified in undergraduate and graduate courses treating linear algebra, numerical analysis, control theory, etc. In nonacademic settings L-A-S can serve as a "desk calculator" for quick solution of any problem which may be decomposed into a sequence of existing L-A-S operators.

The computer resources required by L-A-S are rather modest and may be easily adjusted to a particular computer. With proper use of overlays it is possible to run L-A-S on minicomputers with 64 Kbytes of memory. The size of matrices that can be handled by L-A-S is determined by the available memory. In fact, the only limitation is a total amount of storage allocated for all data elements. In the case of 4000 allocated data elements, for example, the user may define either

10 20 x 20 matrices, or
40 10 x 10 " "
4 30 x 30 " "
1000 2 x 2 , if he wishes!

The software is written in Fortran 77 (there is a version in Fortran IV as well) and is designed to be readily installed under any operating system permitting interactive operations. So far, various versions of L-A-S software have been implemented at the following computer systems: DEC-10, DEC-20, VAX/VMS, VAX/UNIX, HP-1000, PDP-11/70, PDP-11/34, CDC-CYBER, UNIVAC-1100, PRIME-750, PERKIN-ELMER-INTER DATA 8/32, in Yugoslavia, Europe, North and South America. An ample on-line help file containing all necessary information about the L-A-S usage is at the disposal of an inexperienced user, if needed.

2. MULTIPLE OPERATOR FEATURE

With each L-A-S operator associated is a corresponding Fortran subroutine performing specific calculation as, for example, matrix addition, Riccati equation solution, singular value decomposition, etc. The L-A-S operator bears the mnemonic name of the associated subroutine. In above cases the operator names are: + ; RIC and SVD, respectively. L-A-S operators are executed via single-line commands referred to as L-A-S operator statements. L-A-S operator statement are executed one at a time. The statements are interpreted by a part of software referred to as L-A-S interpreter.

After accepting operator statement, L-A-S interpreter checks statement syntax and semantics. In case of an error, an appropriate message is issued on the user's terminal. If all checks are satisfied /1/ control is transferred to the corresponding subroutine. The subroutine, using specified input data, performs the desired calculations and returns control back to L-A-S interpreter. Normally, each L-A-S operator statement refers to a single L-A-S operator. However, the most recent modification allows more than one operator in a single operator statement. This feature is particularly useful for more experienced users who often complain about too many statements which have to be entered.

In order to clarify this feature consider a simple example of calculating the matrix defined by:

$$L_n = A_{22}^{-1} \left[L \; (A_{11} - A_{12} \, L) + A_{21} \right] \qquad (1)$$

assuming that all matrices on the right hand side are given and defined. This problem occurs in block diagonalization /3/ of a given (n x n) matrix A, i.e. in calculating the similarity transformation T which transforms A into H so that:

$$A = \left| \begin{array}{c:c} A_{11} & A_{12} \\ \hdashline A_{21} & A_{22} \end{array} \right| \; ; \quad T = \left| \begin{array}{c:c} L & 0 \\ \hdashline L & I \end{array} \right| \; ; \quad H = \left| \begin{array}{c:c} H_{11} & A_{12} \\ \hdashline 0 & H_{22} \end{array} \right| \text{ and } H = T \, A \, T^{-1} \qquad (1.a)$$

The parse tree of eq. (1) is given in Fig. 1. In order to calculate matrix L_n it is necessary to perform 6 matrix operations in the order specified in Fig. 1. The conventional approach /1/ in calculating matrix L_n would require six operator statements each performing a single matrix operation. By this "classical" approach the matrix L_n might be calculated by a sequence of "single" L-A-S operator statements given by Fig. 2.

However, the recently introduced feature of L-A-S multiple-operator statements permits calculation of matrix L_n by only one L-A-S statement of the form:

A22(INV),L,A11,A12,L(*)(-)(*),A21(+)(*)=LN (2)

In processing such a multiple operator statement, the L-A-S interpreter applies parsing technique /4/ and decomposes the statement (2) into a sequence of single operator statements similar to that in Fig. 2. From (2) it may be concluded that each operator together with the required number of input variables may be regarded as a generalized variable appearing as an input variable in another operator. For example, the variables A12 and L, together with the multiplication operator (*), i.e. the sequence A12,L(*) represents the generalized variable, say T2, which, together with variable A11, enters into the subsequent subtraction operator "(-)" making the sequence A11,T2(-). Analogously, the obtained sequence A11,T2(-) is regarded as another variable, say T3, which together with variable L and the next operator "(*)"forms another generalized variable T4, etc.

Similarly, the "ternary" Riccati operator "RIC" requiring three variable names in input is defined by:

A,Q,S(RIC)=K

where $S=B\ R^{-1}\ B^T$, while K is the Riccati gain matrix. Assuming that the matrix pairs $\{A,B\}$ and $\{Q,R\}$ specifying the system to be optimized and performance index to be minimized, respectively, are given, the corresponding L-A-S multiple operator statement is of the form:

A,Q,B,R(INV)(*),B(T),(*)(RIC)=K (3)

where "T" is the operator name for matrix transposition.

3. L-A-S OPERATOR STATEMENT SYNTAX

From previous examples it may be concluded that the general structure of the operator statement is as follows:

label : input-field (operator-field) = output-field (4)

The terminal symbols " : " , " (" and ")= " are used as field delimiters. The label, which is optional, is used in conjunction with a particular class of operator statements, referred to as program control operators for realization of iterative and recursive calculations. The use of labels is explained in Section 4. The operator field contains the mnemonic name of the subroutine associated with the operator. In the operator field it is possible to specify if the results of the operator calculation are to be displayed immediately on the terminal, or printed on the line printer. The output-field contains the list of variables where the results are to be stored. The input-field contains the list of both variables and generalized variables to be used by the operator. The generalized variable, on the other hand, is structured as:

input-field (operator-field) (5)

The required number of input variables is determined by the operator type, i.e. for unary, binary and generally n-nary operators, the required number is 1, 2 and n, respectively.

The variables appearing both in input and in output fields share a common region in computer memory. Thus, variables specified in the output fields may be referenced in input fields of any operator statement to be executed later on.

To a certain extent the L-A-S language resembles other existing interactive languages such as APL, MATLAB, PROTRAN, MATRIX, CLADAP /5/, /6/, /7/ and the others. All these languages are interactive and accept single-line commands or statements, process them immediately and display the results. One of the major differencies lies in the statement syntax. While majority of existing languages /5/, /6/ attempt to use infix notation, at least for binary operations as addition, multiplication, etc, in the L-A-S language syntax design another approach has been used. Namely, it is well known that both post-fix and pre-fix notations, in contrast to conventional infix notation, are better suited to accommodate general n-nary operators, where n, number of input arguments, might be 0, 1, 2, 3, and so on. We have experienced that from the user's point of view it is not convenient to use different syntax rules for different operator types. The important point is to consider the operator (regardless of whether unary, binary of n-nary, with n 2) as a transformation operating on input data and producing output results. This attitude has consequently been adopted in the L-A-S language syntax.

4. EXAMPLES OF THE L-A-S LANGUAGE USE

In order to illustrate other features of the L-A-S software consider once more the example treated in conjunction with the multiple operator statements. In /3/ it has been shown that in order to perform the block diagonalization of a given (n x n) matrix A, see eq. (1.a), the following nonlinear Riccati type matrix equation has to be solved.

$$A_{22} L - L A_{11} + L A_{21} L - A_{21} = 0 \qquad (6)$$

The submatrices H_{11} and H_{22} defining the transformed matrix H are given by

$$H_{11} = A_{11} - A_{12} L \quad \text{and} \quad H_{22} = A_{22} + L A_{12}. \qquad (7)$$

Equation (6), under the conditions specified in /3/, may be solved by the following iterative process:

$$L_{i+1} = A_{22}^{-1} (L_i H_{11} + A_{21}) \; ; \; \text{with } L_0 = A_{22}^{-1} A_{21} \qquad (8)$$

The formulated problem may be solved by the L-A-S program, i.e. by the sequence of L-A-S operator statements given in Fig. 3.

As it has already been explained, by the statements 1 and 2 the user may define dimensions and all elements of the matrix A, as well as to enter scalars:

- M representing the order of matrices A_{11} and H_{11};
- EPS, representing a sufficiently small positive number used as stopping condition in iterative solution of eq. (8).

Other statemetns perform the following:

3. (m x m) zero matrix, used in statements 5 - 7 is defined.
4. This statement permits the modification of any elements in variables defined so far, for example in matrix A and scalar M. Note that this statement has the label "A".
5. "CTC", Cut by columns operator, defines matrices X and Y satisfying $A = | X \vdots Y |$. Number of columns in X corresponds to the number of columns in Z.
6. "CTR", Cut by row operator partitions matrix X into A11 and A12.

7. "CTR" operator partitions matrix Y into A21 and A22.

8. Matrix A2I = A_{22}^{-1} is defined.

9. Matrix L = A2I $*$ A21 is defined.

10. Matrix H_{11} defined by (7) is calculated. The statement has label "B".

11. Multiple operator statement defining matrix T. Matrix T corresponds to L_{i+1} in (8).

12. NRR - Norm operator. The norm of the matrix difference L-T is determined.

13. MCP - Matrix copy operator. Matrix T is copied into L.

14. IFJ - If Jump operator. This operator performs conditional jump to the statements having labels:

B if $E - EPS < 0$,
B if $E - EPS = 0$,
C if $E - EPS > 0$.

This statement belongs to the class of "program control statements" mentioned before. By this type of statements it is possible (as in Fortran by IF and GO TO statements) to realize recursive and iterative computational processes.

15. The matrix H22 correspons to H_{22} given by (7).

16. RDI - Row dimension operator. The string A(RDI) is a generalized variable corresponding to the number of rows in matrix A. The statement 16 defines k = n - m, n being the order of A.

17. This multiple operator statement defines transformation matrix T given by (5). The parse tree of this statement is given in Fig. 7. The only operators not mentioned and used in this statement so far are.

CTI - Column tie and RTI - Row tie.

Both these operators are binary. In the cases of statements

A,B(CTI)=X and A,B(RTI)=Y

matrices X and Y would become $X = \left| \begin{array}{c:c} A & B \end{array} \right|$ and $Y = \left| \begin{array}{c} A \\ \hdashline B \end{array} \right|$, respectively.

18. OUT operator. Obtained matrices H_{11}, H_{22} and T are displayed on the terminal screen.

After performing all calculations, i.e. after executing statement 18, the user may transfer control back to the statement with label "A" by entering the command: JUMP,A. This type of commands is explained in Section 6.

The statement with the label "A" is the (INP)= operator statement, see Fig. 3. Thus, this statement will prompt the user to specify names of matrices he wants to modify. In this particular case by specifying the name of matrix A, any element, row, column or main diagonal may be modified in A without retyping the whole matrix. After entering desired modifications the whole calculation, i.e. block diagonalization of newly defined A will be performed again.

As it will be explained later, by entering the command

WPF, ⟨block-diag⟩

the whole L-A-S program constituting of the statements from Fig. 3 will be stored on the L-A-S Program file. In this way, the same program may be read back from the program file and executed again without retyping all statements.

In this paper examples give only relatively simple operators. This has been done in order to facilitate understanding of basic features of L-A-S language. A description of all L-A-S operators may be found in

/1/ and /2/.

Creation and use of L-A-S subroutines

The newly introduced feature of user-defined subroutine will be explained on the same example of block diagonalization defined by (5)-(8). If, within a single L-A-S program, the user has to perform block diagonalization of two or more matrices, then the definiton of a subroutine performing this task is a rational solution. To this end, after having the L-A-S program given in Fig. 3, the user has to do the following:

- to eliminate statements number 18, and 1 through 2,
- to include at the beginning the statement of the form A,M,EPS(BDG,SUB)= =H11,H22,T,
- and finally to store the whole program on the L-A-S subroutine file under the name "BDG.SUB",

i.e. file name "BDG" - block diagonalization and file extention "SUB" - user defined subroutine.

The procedures of eliminating and adding statements, as well as storing programs and subroutines on L-A-S program or subroutine files, is explained in Section 6. After performing the specified modifications, the user-defined subroutine will have statements as given in Fig. 4. On the other hand, the main L-A-S program calling this subroutine might be of the following form

```
(INP)=A,B
(DSC)=MA,MB,E
A,MA,E(BDG,SUB)=H11,H22,T
B,MB,E(BDG,SUB)=G11,G22,TB
```

which presumably needs no further explanations.

The important issue in creating user-defined subroutines is that everything what should be done is on the L-A-S language "level", and no Fortran programming is required.

Local and global L-A-S variables

The variables as Z, X, Y, A11, A21, A22, L, NM and E which appear in the subroutine are "local" variables and do not interfere with variables defined in the main program. For example, the variable E from the main program will not share the same location as the local variable E in the subroutine. In fact, during the subroutine execution all local variables will be given the following four-character names:

S01. ; S02. ; S03. ; ..... ; Sij. ; ...

according to the order as they appear, which evidently prevents interference of variables having the same name in the main program and the subroutine.

5. DEFAULT VALUES

It has already been emphasized that operator type (i.e. whether the operator is unary, binary or generally n-nary) determines the number of variables in the input field. However, for some n-nary operators, particulary for those having n=5 or n=6, it is possible to specify a reduced number of variables in the input field. In this case the L-A-S operator assumes some conveniently chosen default values. Some examples of default values are as follows:

a. eps = sufficiently small positive number used in all operators involving rank calculation,
b. frequency range in operators calculating and plotting Nyquist and Bode diagrams,

c. initial guess in iterative solution of algebraic matrix Riccati equation,
d. time interval used in operators calculating and plotting system time responses.

Information on default values may be obtained on-line from the help-file.

6. L-A-S INTERPRETER COMMANDS

The L-A-S software has a number of additional features which make it "user-friendly" i.e. which adapts to the user's experience. This is accomplished via another type of statements, referred to as interpreter commands.

One group of interpreter commands sets the environment most appropriate in relation to user's experience. For example, once the operator statement is entered via terminal keyboard L-A-S interpreter memorizes the statement in the central memory. In this way the whole sequence of operators statement constituting the L-A-S program, may be reexecuted using different input data. If desired, only a part of a stored L-A-S program may be reexecuted. Normally, by default, during reexecution of a L-A-S program, statements are executed sequentially according to thier order in L-A-S program. By the interpreter command TRACE, the interpreter enters into the "trace" mode which means that after each statement the L-A-S program execution halts temporarily. At this moment the user may:

- enter any operator statement to perform some additional calculation, or just display some of the variables,
- enter the interpreter command: JUMP, n, where n is a sequence number of the operator statement to be executed next,
- enter the CONTINUE interpreter command, by which the next operator statement will be executed,
- enter the NOTRACE interpreter command by which the L-A-S interpreter returns to the "no trace" mode, i.e. resumes default sequential execution of all statements in a L-A-S program.

LIST/NOLIST and TYPE/NOTYPE Interpreter commands

During reexecution of L-A-S program, L-A-S interpreter, by default, displays on the terminal screen:

- the operator statement currently executing and
- the results obtained by the operator.

By the interpreter commands NOLIST and NOTYPE, display of operator statements and operator results will be surpassed, respectively.

By the commands LIST and TYPE the interpreter returns back to the default mode of operation.

At each moment during L-A-S program execution, the user may by using interpreter commands STATUS and NAMES, determine the L-A-S interpreter mode or check:

- number of variables defined so far,
- total number of data elements used by these variables,
- names and dimensions of all defined variables.

By the ELMATRIX command any defined variable may be eliminated reducing thus the number of variables as well as total number of data elements presently used.

WPF and RPF (Write/Read Program file) Commands

By the WPF, ❬prog-nam❭ command, L-A-S program, created by entering and executing operator statements, may be stored on the program file. When stored on the program file, L-A-S program is assigned an appropriate name ⟨prog-nam⟩. In this way, more L-A-S programs may be kept on program file and reexecuted when required.

The interpreter command: RPF, ⟨prog-nam⟩ serves for reading L-A-S programs from the program file into central memory. If commands RPF or WPF are entered without specifying the program name, L-A-S interpreter will prompt the user to enter the desired program name.

7. EDITING FACILITIES

By the interpreter command PROGRAM, L-A-S interpreter displays on terminal the L-A-S program currently residing in central memory. Before attempting to reexecute or to store program on the program file, the user may edit some of the program's statements. To this end the following commands are at his disposal:

a. CHANGE,n,abc/xyz	change string of characters
b. ELIMINATE,n,m	eliminate statements
c. INCLUDE,n, ⟨any-oper.-statement⟩	include a statement
d. GCHANGE,n,m,nam1/nam2	global change of variable names

These commands perform the following:

a. The character string "abc" in the operator statement having the sequence n will be substituted by the string "xyz".
b. The operator statements having sequence numbers between n and m will be eliminated from L-A-S program.
c. The statement ⟨any-oper.-statement⟩ will be included in the current L-A-S program immediately after the operator statement with the sequence number n.
d. In the part of the current L-A-S program determined by sequence numbers n and m, all appearences of the variable with the name nam1 will be changed to the new name nam2.

Finally, all necessary information on any aspect of the L-A-S use may be obtained on-line from the help file by entering the command

HELP,xyz

where "xyz" is a mnemonic name of the required information.

8. OTHER FEATURES CONTRIBUTING TO THE SOFTWARE FLEXIBILTY

The whole software is extremely modular so that adding new operator is rather straightforward procedure which may be done by any moderately experienced programmer.

It has already been stated that an operator statement, defined by (3), contains operator mnemonic name and names of both input and output data. However, an interesting new L-A-S language feature is that if either input or output data are not specified within the operator statement, L-A-S interpreter will prompt user to specify which data are to be used and where the results are to be stored. Moreover, even the operator mnemonic name may be omitted, if desired. In this sense the following sequence of "operator" statements

```
( )=
( )=
 .
 .
 .
( )=
```

is a syntactically correct L-A-S program, which might be considered as a "generic" L-A-S program by which any problem could be solved although in an extremly tedious fashion. Executing L-A-S programs of this type converts the whole L-A-S software into the so-called "question-and answer" mode. In this case, in order to obtain necessary information for operator execution, L-A-S interpreter will prompt the user to enter:

- operator name,
- names of input variables and
- names of output variables.

Of course, this mode might be convenient only for beginners. After a litle experience, however, the user will get tired of answering so many question of the same type. This feature has only been mentioned here to illustrate the range of possibilities by which L-A-S software is adjustable to the user's experience and habits.

9. CONCLUSIONS

Interactive computing is a powerful tool in solving control system analysis and design problems. Using an interacitve CAD package the user may obtain solution in a few hours without relying on programmers as intermediaries. The solutions are well documented in terms of procedures, listings, graphs and numerical data. This facilitates that the entire calculation may be repeated and some additional analysis performed.

The L-A-S language described in the paper is a contribution to efforts of control people in obtaining a computational aid for solving various analysis and design problems.

10. REFERENCES

/1/ S. P. Bingulac, N. Gluhajić, "Computer aided design of control systems on mini computer using L-A-S language", IFAC Symposium on CAD of Multivariable Technological Systems, Purdue University, USA, Sept. 15-17, 1982.

/2/ J. H. Chow, S. H. Javid, H. R. Dowse, "User's manual for L-A-S language", System Dynamics and Control Group, GE, January 1983.

/3/ P. Kokotović, "A Riccati equation for block diagonalization of ill-conditioned systems", IEEE Trans. on AC, AC-20, 1975.

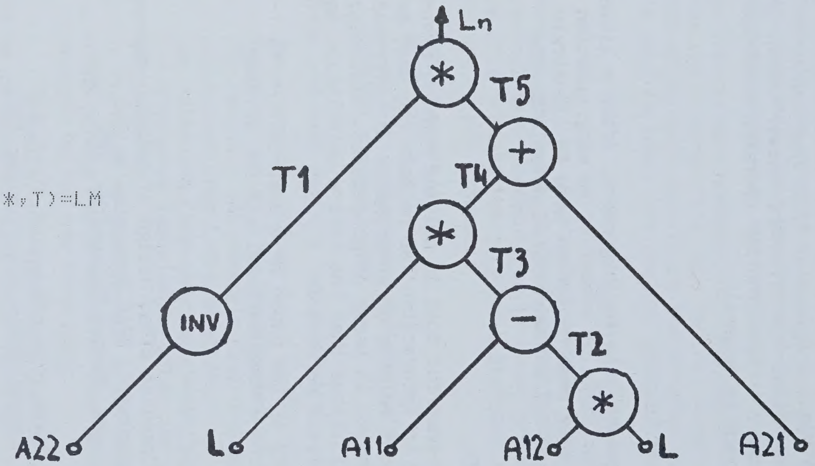
/4/ A. V. Aho, J. D. Ullman, "The theory of parsing, translation and compiling", Prentice Hall, 1973.

/5/ T. J. Aird, J. R. Rice, "PROTRAN Problem solving software", IMSL Tech. report No. 8201, 1982.

/6/ G. Leininger, (editor), Preprints, IFAC Symposium on CAD of Multivariable technological systems, Purdure University, Sept. 15-17, 1982.

/7/ IEEE Control System Magazine, Special issue on Computer aided control system design (CACSD), Decembar 1982.

```
;Single_operator_version
A22(INV)=T1
A12,L(*)=T2
A11,T2(-)=T3
L,T3(*)=T4
T4,A21(+)=T5
T1,T5(*)=LS
;Multiple_operator_version
A22(INV),L,A11,A12,L(*)(-)(*),A21(+)(*,T)=LM
```



```
 1      (INP)=A
 2      (DSC)=M,EPS
 3      M,M(DZM)=Z
 4    A:(INP)=
 5      A,Z(CTC)=X,Y
 6      X,Z(CTR)=A11,A21
 7      Y,Z(CTR)=A12,A22
 8      A22(INV,T)=A2I
 9      A2I,A21(*)=L
10    C:A11,A12,L(*)(-)=H11
11      A2I,L,H11(*),A21(+)(*)=T
12      L,T(-)(NRR,T)=E
13      T(MCP)=L
14      E,EPS(IFJ,T)=B,B,C
15    B:A22,L,A12(*)(+)=H22
16      A(RDI),M(-)=K
17      M,M(DIM),M,K(DZM)(CTI),L,K,K(DIM)(CTI)(RTI)=T
18      H11,H22,T(OUT)=
```

```
 1      A,M,EPS(BDG,SUB)=H11,H22,T
 2      M,M(DZM)=Z
 3      A,Z(CTC)=X,Y
 4      X,Z(CTR)=A11,A21
 5      Y,Z(CTR)=A12,A22
 6      A22(INV,T)=A2I
 7      A2I,A21(*)=L
 8    C:A11,A12,L(*)(-)=H11
 9      A2I,L,H11(*),A21(+)(*)=T
10      L,T(-)(NRR,T)=E
11      T(MCP)=L
        E,EPS(IFJ,T)=B,B,C
      B:A22,L,A12(*)(+)=H22
        A(RDI),M(-)=K
        M,M(DIM),M,K(DZM)(CTI),L,K,K(DIM)(CTI)(RTI)=T
```

EXPERIENCE IN APPLYING A CONTROL DESIGN TECHNIQUE TO A POWER PLANT CONTROL PROBLEM

A.S. BROWER, S.H. JAVID, J.R. WINKELMAN
General Electric Company, Schenectady, New York

ABSTRACT

This paper describes an approach for designing a multi-variable output feedback control based on well known linear quadratic regulator (LQR) theory. It has been applied to design controls for a number of applications ranging from wind turbine-generators to HVDC systems. An example applying this technique to the gas plant portion of a coal gasification plant is presented.

INTRODUCTION

Power plant models tend to be of large order (20th order and more), to be nonlinear, to contain dynamics on multiple time scales, and have limited measurements available. In addition, control problems for these power plants involve multiple, sometimes conflicting, control objectives.

A technique based on a suboptimal linear quadratic regulator [1] has been used to design controllers for these plants. Related work [2] has treated theoretical aspects of multi-variable control assumed in this paper. In addition, [3] introduced a sequential stabilization method similar to control design process used here.

This paper begins with a discussion of the control design technique, followed by a description of a gasification plant model and the results of applying the control design process to this plant.

CONTROL DESIGN TECHNIQUE

The first step in the control design is the linearization of the non-linear power plant model. The resulting model takes the form

$$\dot{x} = Ax + Bu \tag{1a}$$

$$y = Cx. \tag{1b}$$

The states, controls or inputs, and measurements or outputs are, in general, represented by the n-dimensional vector x, the m-dimensional vector u, and the r-dimensional vector y, respectively.

Controls for the gas plant are the signals that control various water, coal slurry and gas flows. The measurement, or output, vector can be grouped into two sets. The first set contains those measurements to be regulated to setpoint, for instance, drum level or temperature. The second set includes additional measurements fed back to assist in meeting the operating objectives and maintaining variables within their constraints. In addition to these measurements, integrals of the errors between some of the measurements to be regulated to setpoint and their setpoints are appended to the plant and introduced as measurements. The result of this grouping should be one control for each measurement to be regulated as needed [2].

Control Performance Specification

Once the model has been generated and possible measurements and controls determined, the control specification should be developed. The specification includes the setpoints, allowable steady-state error, maximum allowable overshoot and operating constraints.

Control Structure Specification

Control structure specification is closely related to linear model analysis. In addition to regulation, each control can assist in minimizing system dynamic oscillations. The purpose of control structure is to specify which controls should be used for each regulated variable, and which controls should be coordinated. Based on the dominant time constants and the degree of interaction of the regulated variables (i.e., observability and controllability), the controls to be coordinated and the measurements to be used for each control can be defined.

The important principle used in coordination is that variables to be coordinated should have significant interactions and have dominant time constants in the same time frame. In the gas plant example, one loop is relatively independent of others. This loop was designed as a local loop. Other controlled variables were closely coupled and effectively coordinated using multi-variable control. The resulting control is decentralized. The determination of which control paths should be decentralized plays a major role in the multi-variable control design procedure.

Controls that are to regulate specified variables to setpoint will, in general, give up some transient regulation capability when coordinated. The net gain in total system performance must exceed the losses in performance on a control-by-control basis. It is for this reason that noninteracting control paths were not coordinated.

Based on the degree of success (or difficulty) in obtaining a control design with good performance, it is necessary to add additional measurements (from the second set of measurements previously listed) for coordination. The choice of these measurements relies on engineering judgement gained from time domain plots, controllability and observability tests, and experience. Once the control structure is specified, the control design is begun.

Control Design

The design technique described here has been successfully used in a number of power system and power plant control problems [4], [5].

The foundation of this design method is the well-known linear quadratic regulator problem

$$\text{minimize } J = \frac{1}{2} \int_0^t [x^T Qx + u^T R u] \, d\tau \qquad (2)$$

subject to (1). The solution to this problem is a full-state feedback control

$$u = -G x \qquad (3)$$

where

$$G = R^{-1} B^T M \tag{4}$$

and M is the non-negative definite solution to the matrix Ricatti equation

$$A^T M + MA - MBR^{-1} B^T M + Q = 0 \tag{5}$$

The full-state feedback control cannot be realized in practice since not all states are measurable. The Medanic output regulator theory [1] is used to map the full-state feedback control (3) into an output feedback control

$$u = Ky \tag{6}$$

by selecting and retaining r of the eigenvalues and eigenvectors from the optimal full-state feedback solution. If X_r are the r right eigenvectors retained, then

$$K = - R^{-1} B^T M X_r(CX_r)^{-1} \tag{7}$$

The closed loop system response is then

$$\dot{x} = (A + BKC)\ x \tag{8}$$

The output feedback solution retains a portion of the eigenspace of the full-state feedback performance to yield a coordinated output feedback control.

Each stage in the design sequence is devoted to one of the decentralized blocks determined in the structural specifications. To illustrate this sequential design process [3], the linear model (1) may be partitioned as follows:

$$\dot{x} = Ax + B_1u_1 + B_2u_2 + B_3u_3 \tag{9}$$

$$y_1 = C_1x \tag{10a}$$

$$y_2 = C_2x \tag{10b}$$

$$y_3 = C_3x. \tag{10c}$$

As a result of control system structuring, the desired controls will take the form

$$u_i = K_iy_i \quad i = 1,2,3.$$

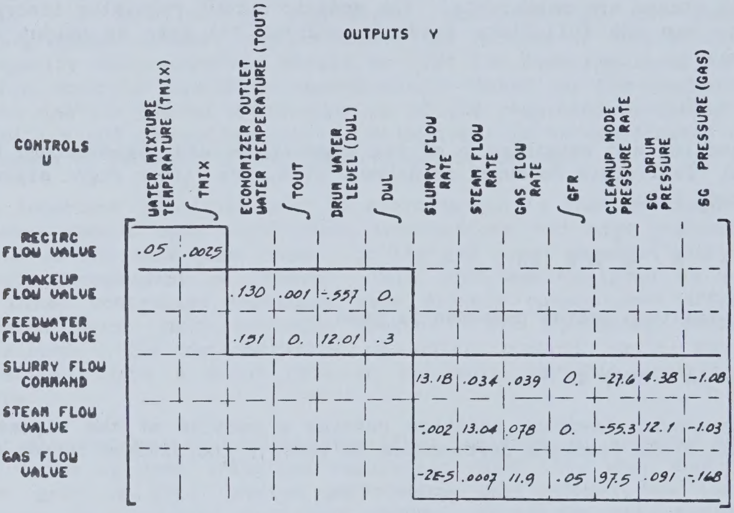
Figure 1 illustrates the end result of such a design process. In the first stage, the output regulator $u_1 = K_1y_1$ is designed using equations (9), (10a), and (2)-(7). This control is then included in the model to yield a new model

$$\dot{x} = (A + B_1K_1C_1)x + B_2u_2 + B_3u_3 \tag{11}$$

$$y_2 = C_2x \tag{12a}$$

$$y_3 = C_3x. \tag{12b}$$

The next stage is to use the output feedback design process to design $u_2 = K_2y_2$ and so on. Experience has shown that the first pass through a design for `a particular system is the most time consuming. Subsequent designs proceed expeditiously.



OUTPUTS Y

CONTROLS U	WATER MIXTURE TEMPERATURE (TMIX)	∫TMIX	ECONOMIZER OUTLET WATER TEMPERATURE (TOUT)	∫TOUT	DRUM WATER LEVEL (DWL)	∫DWL	SLURRY FLOW RATE	STEAM FLOW RATE	GAS FLOW RATE	∫GFR	CLEANUP MODE PRESSURE RATE	SG DRUM PRESSURE	SG PRESSURE (GAS)
RECIRC FLOW VALVE	.05	.0025											
MAKEUP FLOW VALVE			.130	.001	-.557	0.							
FEEDWATER FLOW VALVE			.151	0.	12.01	.3							
SLURRY FLOW COMMAND							13.18	.034	.039	0.	-27.6	4.38	-1.08
STEAM FLOW VALVE							-.002	13.04	.078	0.	-55.3	12.1	-1.03
GAS FLOW VALVE							-2E-5	.0007	11.9	.05	97.5	.091	-.168

Figure 1 Gas Plant Control Gain Matrix (K Matrix)

Each stage of the control design is checked at other operating points by observing the eigenvalues. Since the linear models are obtained from the nonlinear simulation model, this linear sensitivity test examines the effectiveness of design for small variations.

After the design is checked on linearized models, the control is simulation tested on the nonlinear model for several different input disturbances. Nonlinear simulation tests can be classified in two categories: "friendly" and "unfriendly." The friendly test is one that perturbs the plant evenly to a new operating point. An unfriendly test is one that subjects the plant model to a severe disturbance. Such unfriendly tests are often the only way to determine if a given control design rigorously meets all the control system specifications.

POWER PLANT CONTROL APPLICATION

The power plant model used for illustration of the control design method is based on a coal gasification combined cycle plant of typical design. Figure 2 shows a block diagram of the gas plant only, including the components, the gas and steam/water flow paths, and the six major control points.

The gas flow path in Figure 2 is as follows. A coal/water slurry is mixed with oxygen and is gasified in an entrained flow reactor under pressurized slagging conditions. The oxygen/slurry ratio is controlled by a local controller such that this ratio never exceeds a fixed value. Since both the slurry and the oxygen follow the slurry flow command, the coal/water slurry and oxygen valves represent a single control point, the slurry flow command.

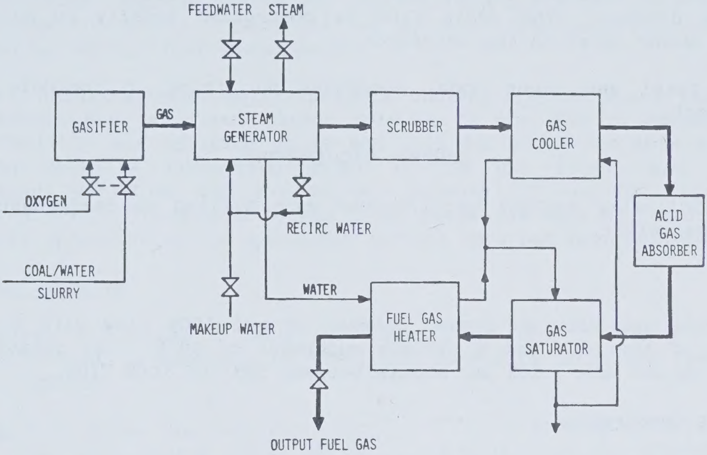


Figure 2 Gas Plant Block Diagram

The hot gas from the gasifier is cooled in the steam generator (SG). The gas is then washed and cooled in a scrubber by a water flow not shown in Figure 2 and then further cooled in a gas cooler.

The cool gas is cleaned in an acid gas absorber, which selectively removes sulfur from the raw gas. The cooled, clean gas is heated and saturated with water vapor in a gas saturator, and then is further heated in the fuel gas heater, which also superheats the water vapor. The output fuel gas flow is controlled by the gas flow valve.

Although the model includes many temperature calculations along the gas flow path, only three equivalent gas pressures are calculated, using the perfect gas laws. These three pressure nodes are separated by two equivalent flow resistances.

Figure 2 also shows the steam/water flow path. On the steam/water side, the steam generator is a forced circulation, saturated steam boiler, with constant circulation flow. Preheated feedwater is mixed with economizer exit water and the mixed feedwater is supplied to the drum. Some of this water is boiled in the SG; this saturated steam flows from the drum through the steam flow control valve. The preheated feedwater flow (which comes from the Heat Recovery Steam Generator elsewhere in the power plant) is controlled by a valve to maintain constant SG drum water level and is assumed available on demand. The economizer inlet water is a mixture of saturated water recirculated from the SG drum and cold makeup flow. The drum water flow is controlled by the recirculation flow valve to maintain constant economizer inlet water temperature (TMIX). The cold makeup flow is controlled by the makeup flow valve to maintain economizer exit water temperature at setpoint (TOUT) and is assumed available on demand.

Some of the economizer exit water flow is diverted to the fuel gas heater to superheat the gas; a local controller diverts the right amount of flow to maintain a constant temperature in the first water-side section of the fuel gas heater. The water leaving the fuel gas heater goes to a circulation loop between the gas saturator and the gas cooler. Most of this water is absorbed in the gas in the saturator; the remainder of the water is drained. The drain flow is controlled locally to maintain a constant water level in the saturator.

The total gas plant model comprises 60 states, 6 controls, and 9 measurements.

CONTROL REQUIREMENTS

The following control requirements were defined as performance goals for the gas plant.

Overall

Produce gas flow on demand between 50 and 100% flow with a minimum pressure of 400 psia and a minimum superheat of 50°R. Gas demand change rates to be met are ± 10% per minute between 50% and 100% flow.

Component Constraints

1. Economizer Inlet Water (TMIX) within ± 5°R of Setpoint.
2. Economizer Exit Water (TOUT) within ± 50°R of Setpoint.
3. Saturator Pressure shall not decrease at a rate greater than 5 psi/minute.
4. Drum Steam Pressure: $\leq$ Relief Valve Setting.
5. Drum Water Level: -5 in. $\leq$ DWL $\leq$ $+5$ in.

RESULTS

Gas Plant Control

A decentralized, multi-variable control system was successfully designed for the gas plant. The gain matrix, K, for this control system is shown in Figure 1. (The gain values in this matrix are dimensional and cannot be directly compared with each other.) Figure 1 also defines the six controls and nine measurements plus integrals of four of the nine measurements. All of the measurements are actually errors from setpoints, except the cleanup node pressure rate, which is a high pass filtered output and is non-zero only if negative (pressure decreasing).

The three decentralized blocks illustrate the process of determining the control structure. The recirculation flow valve regulates the water mixture temperature and the makeup flow valve and feedwater flow valve jointly regulate the economizer output water temperature and drum water level. Since a change in position in any of these control valves will result in a change in all three regulated variables, one might think that all three valves should be coordinated. However, due to the physical proximity of the water mixture temperature sensor to the recirculation flow valve, the dominant time-constant associated with its regulation to setpoint is much smaller than those related to the other variables to be regulated. Consequently, control of the recirculation flow was not coordinated with the makeup and feedwater flow controls, as Figure 1 illustrates.

The third control block shown in Figure 1 is associated with control of the energy balance in the gas plant. The steam flow and gas flow rates represent output flows and the slurry flow rate represents an input flow. This control structure allows for good plant maneuverability. The last three measurements were brought into this coordination in order to bring the controlled response of the gas plant within the constraints set in the control specification. The transient results illustrate how control was improved by adding these measurements.

This decentralized, multi-variable control system meets the control requirements for gas flows varied between 70 and 100% of rated. Closed loop stability was verified at 70 and 100% gas flow (the gain matrix was designed using a linearized model at 70% gas flow), and plots of significant variables for a transient between 100 and 70% gas flow are presented next. Transients to gas flows below 70% were not explored since the basic objective of the gas plant control work had been achieved.

Transient Results

Figures 3 through 7 show the results of the "friendly" nonlinear simulation test, a maneuver in which gas demand is decreased from 100% to 70% at -10%/min.

Figure 3 shows the regulation of steam, gas and slurry flow on a trajectory leading to the new 70% operating point. The deviations from the ramped setpoint result both from the inherent plant response limitations and the three additional coordinated variables (cleanup node pressure rate, SG drum pressure and SG gas pressure) that were introduced to assist the control in holding the plant response within constraints. Figure 3 shows that as the transient passes, these additional variables play a lesser role, and the flows are regulated.

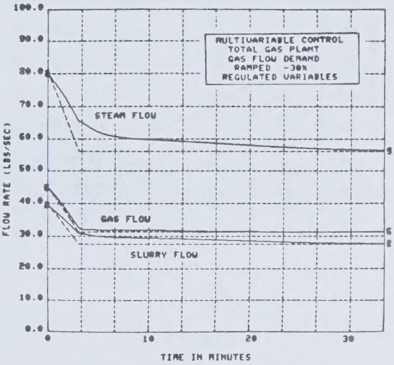


Figure 3

Figure 4 shows the results from control of the water mixture temperature (TMIX), the economizer outlet water temperature (TOUT), and the drum water level. The maximum mixture water temperature error (~|0.35°R|) is well within the ±5°R specification because the control is very fast. The outlet water temperature has a maximum error (~|25°R|), one-half that specified. The drum level error (~|0.3 in|) is also well below the ±5 inch specification.

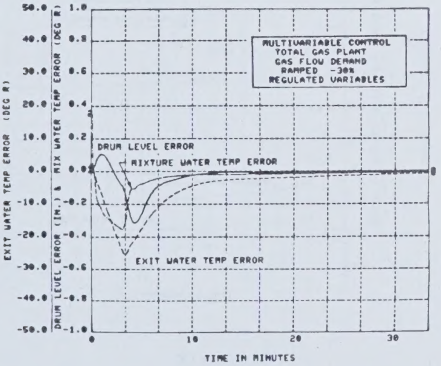


Figure 4

Figure 5 shows the response of the three nodal pressures that result from the coordinated flow rate controls. Their response is stable and within constraints.

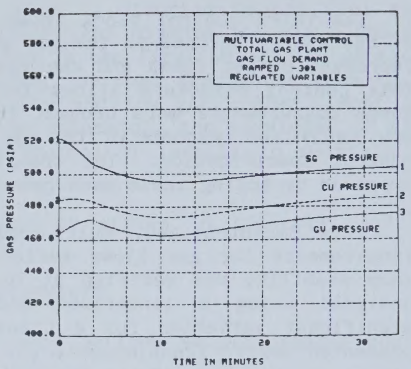


Figure 5

Figure 6 shows the effect of omitting the SG gas pressure measurement in the control of slurry, gas and steam flows. The pressures in Figure 6 vary much more than those shown in Figure 5. Omitting the other nonregulated variables in the energy balance control block results in equally unacceptable state excursions.

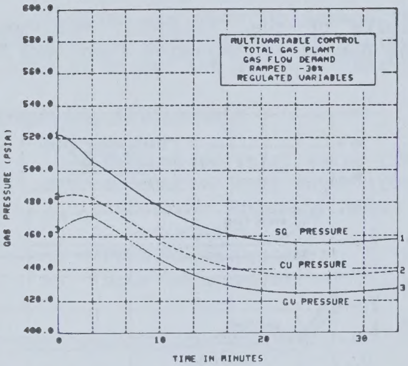


Figure 6

Figure 7 shows the controlled water flows that result from the control design of Figure 1 during the "friendly" test, and corresponding to Figures 3-5.

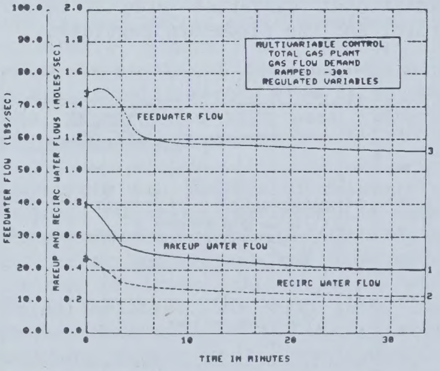


Figure 7

Figure 8 illustrates the regulation of the steam, gas, and slurry flows during an "unfriendly" test in which the gas plant demand is suddenly stepped from 100% to 70%. Although all the variations in the plant are larger than those for the same demand change of -10%/min, only one constrained variable exceeds limits. The cleanup node pressure rate reaches -0.19 psi/sec (-11.4 psi/min) compared with the limit of -0.083 psi/sec (-5 psi/min).

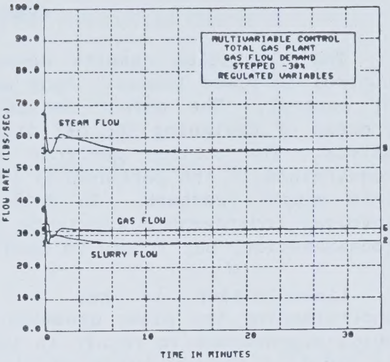


Figure 8

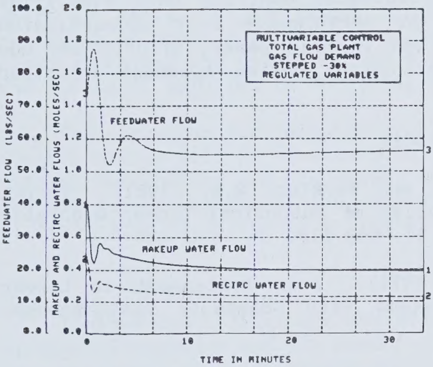


Figure 9

Figure 9 shows the variations in the controlled water flows during the same "unfriendly" test.

CONCLUDING REMARKS

The simulation results demonstrate the effectiveness of coordinated control in power plants. Many methods could be used to achieve this type of control. The output design approach used here [1] simplifies the process of designing the off diagonal, coordination terms in the feedback matrix. The control designer weights variables to be regulated and/or constrained in the performance index and then maps the full state feedback into output feedback. To date, this process of weighting and mapping involves considerable knowledge of the dynamics of the power plant under consideration, but it can be used effectively.

Linearization is key to both the control design itself and understanding the plant dynamics. In particular, the process of choosing which eigenvalues to retain in the output feedback projection (Equation 7) is simplied through study of the eigenvectors of the open loop plant and the closed loop plant resulting from full state feedback (Equations 1 and 3).

This approach successfully demonstrates that the linear quadratic regulator can provide the foundation for an implementable control design without resorting to the more complex Kalman Filters or Observers. In addition, the use of decentralized coordinated controls both simplifies the design process and improves plant performance over uncoordinated control. Based on the results described in this paper, efforts are now being focused on application of this control design approach to other power plant control problems.

REFERENCES

[1] Hopkins, W.E., Jr., Medanic, J., and Perkins, W.R. (1981). Output Feedback Pole Placement in the Design of Suboptimal Linear Quadratic Regulators, Int. J. Control (GB), 34, 593-612.

[2] Davison, E.J. and Smith, H.W. (1971). Pole Assignment in Linear Time-Invariant Multi-variable Systems with Constants Disturbances, Automatica (GB), 7, 489-498.

[3] Davison, E.J. and Gesing, W. (1979). Sequential Stability and Optimization of Large Scale Decentralized Systems, Automatica (GB), 15, 307-324.

[4] Murdoch, A., Winkelman, J.R., Javid, S.H., and Barton, R.S. (1983). Control Design and Performance Analysis of a 6 MW Wind Turbine-Generator, IEEE Trans. Pow. Ap. and Sys. (USA), PAS-102, 1340-1347.

[5] Brower, A.S., Javid, S.H., Patel, A.S., Younkins, T.D., Quentin, G.H., "Application of a Coordinated Control Design Method to a Coal Gasification Plant," Preprints of IFAC Workshop on Modeling and Control of Electric Power Plants, Como, Italy, September 1983.

[6] Johnson, C.D. (1968). Optimal Control of the Linear Regulator with Constant Disturbances, IEEE Trans. Autom. Control (USA), AC-13, 416-421.

COMPUTER-AIDED DESIGN OF DIGITAL CONTROL USING PROJECTIVE CONTROLS

N. GLUHAJIĆ
Institute M. Pupin, Beograd, Yugoslavia

J. MEDANIĆ
Department of General Engineering, University of Illinois

D. PETRANOVIĆ
University of Titograd, Titograd, Yugoslavia

ABSTRACT

In this paper a complete procedure for design of static and low order dynamic regulators for multivariable discrete linear systems based on the method of projective controls is presented.

The method of projective controls uses the solution of an optimal LQ problem as a reference solution. In static regulator design the method is based on retention of r - eigenvalues and the r - dimensional optimal subspace spanned by the corresponding eigenvectors of the reference system where r represents the number of measured output variables. If a dynamic regulator of order p is to be used for control it is shown that the (r + p) eigenvalues and the associated (r + p) - dimensional invariant subspace of the reference solution is retained.

The algorithm for computer-aided design of static and dynamic digital controllers as well as computational aspects and the realization of the digital controllers are discussed. Numerical example and the design and implementation results of the digital controller for the force-regulating electrohydraulic actuator system are included to illustrate the procedure.

INTRODUCTION

Digital controllers are now widely used in control. The falling price of digital hardware and the need for the high performance control systems have motivated their increased application. However, design procedures and the realization and testing of digital control system are still tedious and time-consuming so that computer-aided tools for hardware and software design of digital controllers are needed.

In this paper a methodology that is used in the design of digital control systems is presented. Flexible hardware configuration [13] and computer-aided design software were developed and are used in the controller structure selection and parameters determination as well as for real-time testing. The system is also used as the mP (microprocessor) based direct digital controller development system. The structure and the parameters of the controller are obtained using projective controls approach [7-12].

Design of digital control systems has received considerable attention in the literature. Problems have been approached using many different approaches. From the modern control point of view the first approach was through the solution of optimal LQ and LQG problems [1-4]. Practical implementation issues motivated research on the optimal output regulators [5,6]. However, except for the derivation of the associated necessary conditions for optimality, this approach does not provide deeper analytical treatment nor insight into the properties of the resulting closed loop system. The modified formulation of the output regulator problem was proposed in [8-11] for continuous and in [11,12] for discrete systems. This approach is briefly described, since it presents the basis for the complete design procedure given in the paper. Then, the static and low order dynamic digital regulator design procedure, as well as a brief description of the digital controllers development system are given. The design of electrohydraulical digital control system using the computer-aided design procedure is described, as an example.

STATIC CONTROLLER DESIGN USING PROJECTIVE CONTROLS

Consider the linear discrete system

$$x(k+1) = A\,x(k) + B\,u(k),\; x(0) = x_o \tag{1}$$

$$y(k) = C\,x(k)$$

where $x(k)\varepsilon R^n$, $u(k)\varepsilon R^m$, $y(k)\varepsilon R^r$, $A\varepsilon R^{nxn}$, $B\varepsilon R^{nxm}$, $C\varepsilon R^{rxn}$ and R^1 is the 1 - dimensional linear real vector space. Choosing the quadratic criterion

$$J = 0.5 \sum_{k=o}^{\infty} \{x(k)^T\,Q\,x(k) + u(k)^T\,R_o\,u(k)\} \tag{2}$$

and assuming $Q > 0$ and $R_o > 0$, (A,B) reachable and (C,A) detectable the optimal solution to the Linear Quadratic problem is

$$u\;(k) = -\,(R_o + B^TM_c\;B)^{-1}\;B^TM_c\;A\,x(k) = -\,K_o\,x(k) \tag{3}$$

Where M_c is the solution of the algebraic matrix Riccati caustion

$$M_c = A^TM_c\;A - A^T\;M_cB\;(R_o + B^TM_cB)^{-1}\;B^TM_cA + R \tag{4}$$

The minimal value of the quadratic criterion is

$$J = 0.5\;x(0)^T\;M_c\;x(0) \tag{5}$$

and the resulting closed loop system is

$$x(k+1) = [A - B\;(R_o + B^TM_cB)^{-1}\;B^TM_c\;A]\;x(k) = F\,x(k) \tag{6}$$

The reference dynamics is characterized by X and Λ, such that $F\,X = X\,\Lambda$. Let $R\,\{X_r\}$ span an r-dimensional invariant subspace of F, i.e.

$$F\,X_r = X_r\,\Lambda_r \tag{7}$$

It is shown [11] that the projective control

$$u(k) = -\,(R_o + B^TM_c\;B)^{-1}\;B^TM_c\;A\,P\,x(k) \tag{8}$$

where

$$P = X_r\;(C\,X_r)^{-1}C \tag{9}$$

possesses the following properties:

(a) the control is the function of the system output

$$u(k) = - K_o X_r \ (C X_r)^{-1} y(k) \tag{10}$$

(b) the invariant subspace $R\{X_r\}$ is retained from the optimal reference system

$$(A - B K_o P) X_r = A_c X_r = X_r \Lambda_r \tag{11}$$

(c) the quadratic criterion value is given by

$$J = 0.5 \ x(0)^T M \ x(0) \tag{12}$$

where $M = M_c + D$, $D = V^T D_o V$, $D_o = A_r^T D_o A_r + G$, $G = Y^T A^T M_c B . (R_o + B^T M_c B)^{-1} M_c A Y$, with $V X_r = 0$, $C Y = 0$ so that $0.5 \ x(o)^T M \ x(o) = 0.5 \ x(o)^T M_c \ x(o)$ for $x(o) \varepsilon R\{X_r\}$,

(d) the dynamics of the residual subsystem is governed by

$$A_r = V A Y \tag{13}$$

LOW ORDER DYNAMIC CONTROLLER DESIGN

If the spectrum of the closed loop system obtained with static output controller (10) does not satisfy, a dynamic controller can be introduced. Define the dynamic controller of order p and structure

$$z(k+1) = H \ z(k) + D \ y(k) \tag{14}$$

where $z(k) \varepsilon R^p$, $H \varepsilon R^{pxp}$ and $D \varepsilon R^{pxr}$, and the extended system

$$x_e(k+1) = \begin{bmatrix} H & DC \\ 0 & A \end{bmatrix} x_e(k) + \begin{bmatrix} 0 \\ B \end{bmatrix} u(k) = A_e \ x_e(k) + B_e \ u(k) \tag{15}$$

$$y_e(k) = \begin{bmatrix} I_p & 0 \\ 0 & C \end{bmatrix} x_e(k)$$

where $x_e(k) = [z(k)^T x(k)^T]^T$ and $y_e(k) = [z(k)^T y(k)^T]^T$

With the performance criterion

$$J = 0.5 \sum_{k=o}^{\infty} \{x_e(k)^T \ Q_e \ x_e(k) + u(k)^T R_o \ u(k)\} \tag{16}$$

$$Q_e = \begin{bmatrix} 0 & C \\ 0 & Q \end{bmatrix}$$

The solution of the related optimal LQ problem is

$$u(k) = -(R_o + B_e^T M_{ec} B_e)^{-1} B_e^T M_{ec} A_e x_e(k) \tag{17}$$

where M_{ec} is positive definite solution of the algebraic Riccati equation

$$M_{ec} = A_e^T M_{ec} A_e - A_e^T M_{ec} B_e (R_o + B_e^T M_{ec} B_e)^{-1} B_e^T M_{ec} A_e + Q_e \tag{18}$$

It can be shown [11] that the M_{ec} which solves the stated LQ problem is of the form

$$M_{ec} = \begin{bmatrix} 0 & 0 \\ 0 & M_c \end{bmatrix} \tag{19}$$

where M_c is the solution of (4). The closed loop reference system matrix is

$$F_e = \begin{bmatrix} H & DC \\ 0 & F \end{bmatrix} \tag{20}$$

with F given in (6). The closed loop spectrum is defined by

$$F_e X_e = X_e \Lambda_e \tag{21}$$

and can be partitioned as

$$X_e = [X_{p+r} \; X_o], \quad \Lambda_e = \operatorname{diag}(\Lambda_{p+r}, \Lambda_o) \tag{22}$$

$$X_{p+r} = \begin{bmatrix} W_p & W_r \\ U & Y \\ V & Z \end{bmatrix} \qquad \begin{matrix} W_p \varepsilon R^{pxp}; \quad W_r \varepsilon R^{pxr}; \quad U \varepsilon R^{rxp}; \\ V \varepsilon R^{(n-r)xp}; \quad Y \varepsilon R^{rxr} \text{ and } Z \varepsilon R^{(n-r)xr}, \end{matrix}$$

and X_o and Λ_o define the spectrum of the residual system. An (r+p) -dimensional invariant subspace of the reference system can be retained defining the projection matrix

$$P_e = X_{p+r} (C_e X_{p+r})^{-1} C_e \tag{23}$$

and the corresponding projective control

$$u(k) = -(R_o + B_e^T M_{ec} B_e)^{-1} B_e^T M_{ec} A_e P_e x_e(k). \tag{24}$$

In the case the matrix C_e is of the form

$$C_e = [I_{p+r} \;\; 0] \tag{25}$$

it can be shown that the control (24) is

$$u(k) = - (R_o + B^T M_c B)^{-1} B^T M_c A [P_z P_y] [z(k) \;\; y(k)]^T \tag{26}$$

where

$$P_z = \begin{bmatrix} 0 \\ N_p \end{bmatrix}, \quad P_y = \begin{bmatrix} I_r \\ N_r \end{bmatrix} \tag{27}$$

N_p and N_r are defined by the relation

$$[N_p \;\; N_r] = [V \;\; Z] \begin{bmatrix} W_p & W_r \\ U & Y \end{bmatrix}^{-1} \tag{28}$$

The residual system is

$$A_{re} = A_r + B_o P_o A_{12} \tag{29}$$

$$B_o = V - Z Y^{-1} U$$

$$P_o = W_p^{-1} W_r (Y - U W_p^{-1} W_r)^{-1}$$

In (29) only P_o depends on H and D. Once P_o is selected via the solution of the pole-placement problem implied by (29), H and D are given by

$$H = W_p H_o W_p^{-1} \quad H_o = (\Lambda_r - W_p^{-1} W_r \Lambda_r Y^{-1} U) (I - W_p^{-1} W_r Y^{-1} U)^{-1} \tag{30}$$

$$D = W_p D_o \quad D_o = (W_p^{-1} W_r \Lambda_r - \Lambda_r W_p^{-1} W_r) (Y - U W_p^{-1} W_r)^{-1}$$

with W_p arbitrary and implying full freedom in controller representation. Detailed description of the procedure can be found in [9-12].

CONTROLLER DESIGN HARDWARE AND SOFTWARE SUPPORT

In the process of digital controller design and implementation a Digital Controller Development System (DCDS), [14] was set up. Its hardware configuration is depicted in Fig. 1 and it consists of a mini computer system, bus adopter, expanded bus, multiplexer, mini bus, AD and DA convertors. After the controller is designed and tested the system to be controlled is connected to the hardware configuration as in Fig. 2. If the controller structure is defined the controller parameters are sent directly to the mP based digital controller so that tests of the control system performances can be carried out.

The software that is used in the controller design and implementation procedure consists of:

off - line procedures including

- nonlinear model simulation
- linearized model simulation
- simulation procedures for linear model determination via parameter identification procedures
- identification procedures
- controller structure and parameter determination procedures
- finite wordlength effect testing procedures
- communication support software
- graphic facility support software, and

real - time procedures including

- interrupt routines (or equivalent Real Time Clock support routines)
- AD and DA onvertor support routines
- controller realization routines.

The software is interactive and portable. The design of the digital controller using projective controls is done using the LAS language [15].

EXAMPLE

In the following, an example of design and implementation of digital control using projective control will be presented to illustrate the overall procedure. The system to be controlled is schematically given in Fig. 3. It consists of an electrohydraulic servo valve and a hydraulic actuator. The aim of the controller design is to control the force that is developed on the actuator by means of the oil flow control through the electrohydraulical servo valve. The system input variable is the current - i -, and the output variable is the force - F - measured by a force transducer.

In order to derive the controller structure and parameters and determine control system performances the system is connected to the digital controller development system as shown in Fig. 1.

Then a best linear model of third order was identified using ARMA models and the least square estimates of the output error equation [13]. The system input used in identification from input-output pairs was the sum of cosine functions with frequencies of 1, 3, 5, 7 and 11 Hz. The inputs and resulting output variables obtained and used in the parameter identification procedure are shown in Fig. 4. The input and output signals were sampled with a sampling time of 50 m sec.

The obtained third order linear model was:

$$x(k+1) = \begin{bmatrix} 0.31033 & 1. & 0. \\ 0.46722 & 0. & 1. \\ 0.15578 & 0. & 0. \end{bmatrix} x(k) + \begin{bmatrix} 0.07204 \\ -0.1224 \\ 0.058 \end{bmatrix} i(k) \qquad (31)$$

$$F(k) = [1. \;\; 0. \;\; 0.]\, x(k)$$

where $x(k) \varepsilon R^3$ is the state vector, i(k) and F(k) are the input and output variables. The open loop system eigenvalues are

$$\Lambda^o = \{0.96326, -0.32647 +/-j0.23483\}$$

The LQ design with:

$$Q = \mathrm{diag}\ (100\ ,\ 10\ ,\ 10\ ) \qquad R_o = 1$$

produced the optimal control given by (3),

$$i(k) = -\ [0.66858 \quad 3.6857 \quad -1.6929]\ x(k), \tag{32}$$

where the corresponding Riccati matrix equation solution M_c is

$$M_c = \begin{bmatrix} 386.31 & 302.57 & 313.45 \\ 302.57 & 361.68 & 316.2 \\ 313.45 & 316.2 & 363.15 \end{bmatrix}$$

The resulting closed loop system poles were:

$$\Lambda^c = \{0.94597, -0.067248 +/-j0.25090\}$$

Using the described design procedure for dynamic controller design, a controller of the form

$$z(k+1) = 0.05\ z(k) + 1.\ F(k) \tag{33}$$

$$i(k) = -0.649\ z(k) - 0.61373\ F(k)$$

was obtained. The resulting closed loop system eigenvalues are now

$$\Lambda_e = \{-0.42958, -0.067248 +/-j0.2509, 0.96863\}$$

The subspace corresponding to the eigenvalues Λ_2^o and Λ_3^o of the optimal LQ design was retained in the control system with the dynamic controller, as this was the only possibility due to the presence of a complex pair of roots in Λ^o.

Typical simulation result for optimal LQ design and the dynamic control based on projective control is given in Fig. 5, where the force F (variable x (k)) and the corresponding controls are shown.

The digital control was implemented on the IMP - 85 - 83 mP based controller using 16-bit fixed point arithmetic. One of the test results is shown in Fig. 6. Graph No. 1 represents the reference signal, and graph No. 2 the force F obtained from the controlled system.

CONCLUSION

The digital control design using projective controls is presented in the paper. The computer-aided type of procedure is given for the digital controller design and implementation. The example of electrohydraulical system digital controller design is also presented.

REFERENCES

[1] P. Dorato and A.H. Lewis, "Optimal Linear Regulators: The discrete time case," IEEE Trans. on AC, Vol. AC-16, No. 6, Dec. 1970.

[2] H. Kwakernak and R. Sivan, Linear Optimal Control Systems, J. Wiley & Sons, N.Y. 1972.

[3] J. F. Franklin and J. Powel, Digital Control of Dynamic Systems, Addison-Wesley, 1980.

[4] P. Katz, Digital Control using Microprocessors, Prentice Hall, 1981.

[5] W.S. Levine and M. Athans, "On the Determination of the Optimal Constant Output Feedback Gains for Linear Multivariable Systems," IEEE Trans. on AC, Vol. AC-15, No. 1, Feb. 1970.

[6] T.L. Johnson and M. Athans, "On the Design of Optimal Constrained Dynamic Compensator for Linear Constant Systems," IEEE Trans. on AC, Vol. AC-15, Dec. 1970.

[7] J. Medanić, "On Stabilization and Optimization by Output Feedback," XII Ann. Asilomar Conf. on Circuits and Systems, Pacific Grove, CA, Nov. 1978.

[8] J. Medanić, "Design of Low Order Optimal Dynamic Regulators for Linear Time Invariant Systems," 1979 Conf. on Information Sciences & Systems, John Hopkins Univ., March 1979.

[9] W. Hopkins, J. Medanić and W. Perkins, "Output Feedback Pole Placement in the Design of Suboptimal Quadratic Regulators," Int. J. Cont., 34, 1981.

[10] J. Medanić and Z. Uskoković, "The Design of Optimal Output Regulators for Linear Multivariable Systems with Constant Disturbances," Int. J. Cont., Vol. 37, No. 4, 1983.

[11] J. Medanić, D. Petranović and N. Gluhajić, "The Design of Low Order Regulators for Digital Control Systems," submitted for publication.

[12] J. Medanić, D. Petranović and N. Gluhajić, "Design of Static Regulators for Digital Control Systems with One-step Delay Information Structure," submitted for publication.

[13] K.J. Astrom and P.E. Ewkhoff, "System Identification - A Survey," Automatic, 7, 1971.

[14] S. Ivančević, N. Vukičević and N. Gluhajić, Design of Digital Controller IMP-DDC-85-83, (in serbo-croatian), Institut Mihailo Pupin Internal Report, Beograd, Yugoslavia, 1983.

[15] S.P. Bingulac and N. Gluhajić, "Computer Aided Design of Control Systems on Mini Computers Using LAS Language," IFAC Symposium on Com. Aided design of Multivariable Techn. Syst., Sept. 1982, Purdue Univ., In., USA.

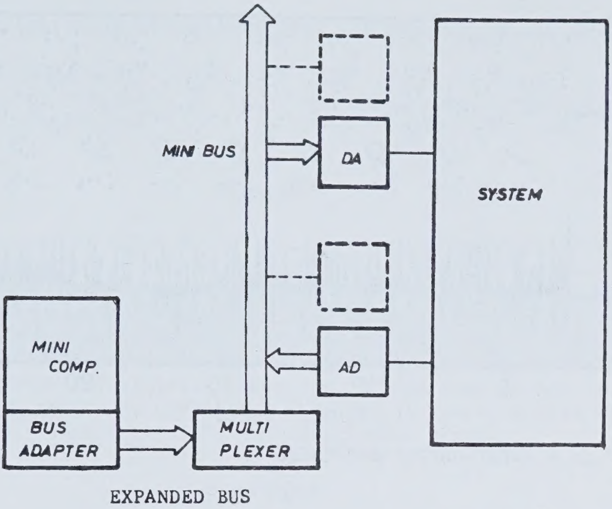

MINI BUS
DA
SYSTEM
MINI COMP.
AD
BUS ADAPTER
MULTI PLEXER
EXPANDED BUS

Fig. 1

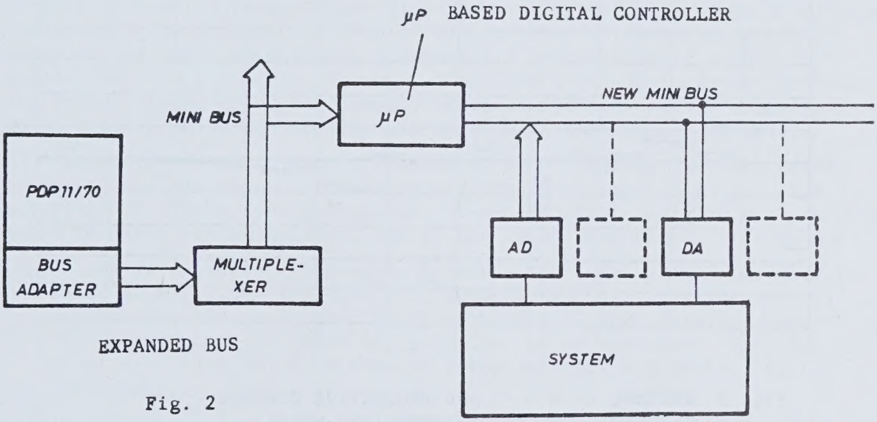

μP BASED DIGITAL CONTROLLER
MINI BUS
μP
NEW MINI BUS
PDP 11/70
BUS ADAPTER
MULTIPLE-XER
AD
DA
SYSTEM
EXPANDED BUS

Fig. 2

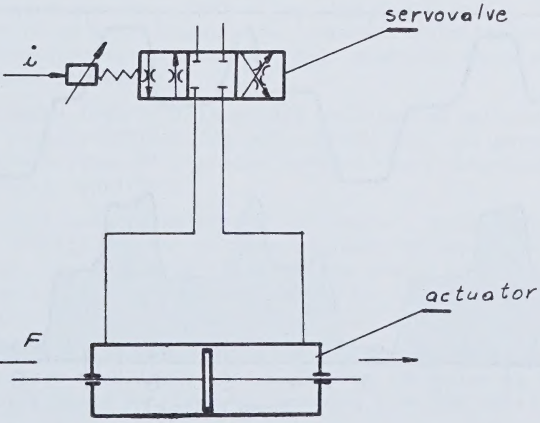

servovalve
i
actuator
F

Fig.3

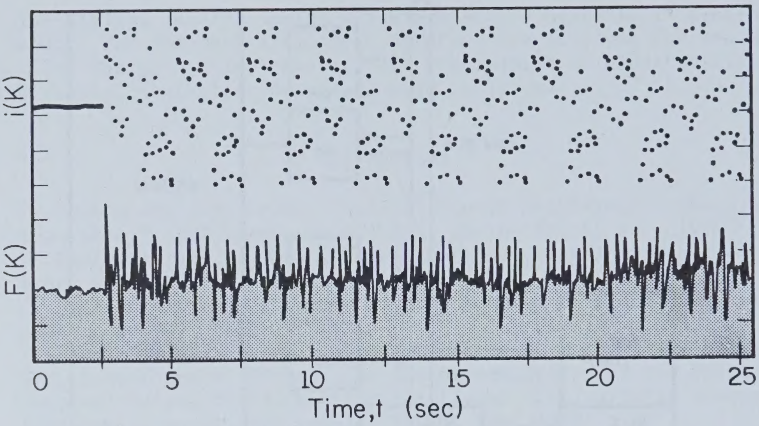


Fig. 4 INPUT-OUTPUT SEQUENCES

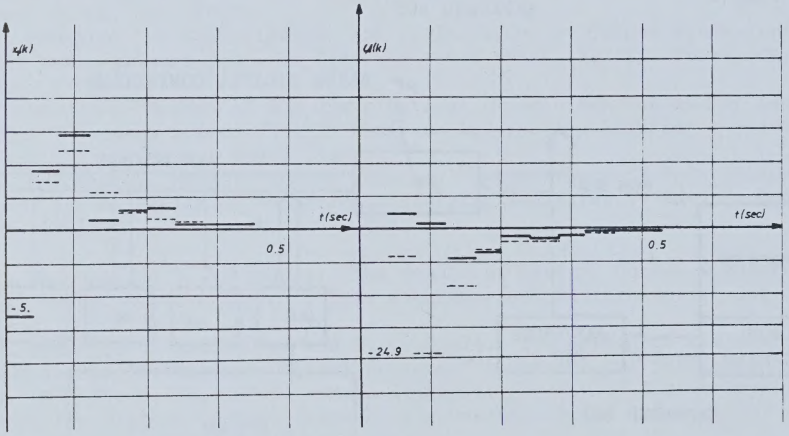


Fig. 5 OPTIMAL (- - - -) AND PROJECTIVE CONTROL (———) INDUCED RESULTS FOR LINEARIZED MODEL

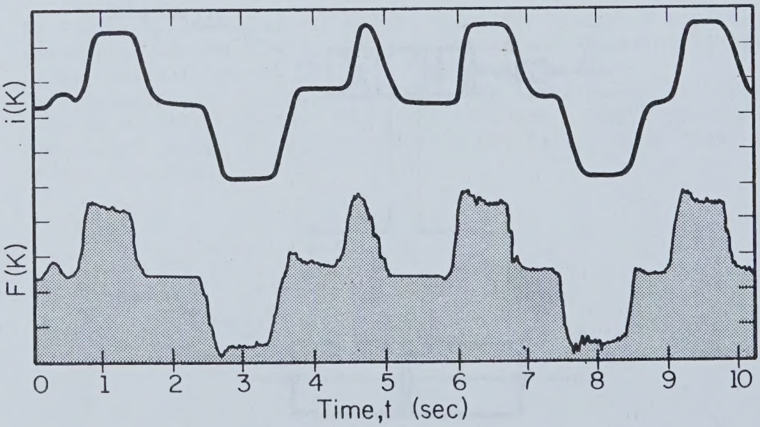


Fig. 3 NONLINEAR SYSTEM SIMULATION RESULTS

COMPUTER-AIDED DESIGN OF CONTROL SYSTEMS: AN EXAMPLE OF WORKING SOFTWARE

S. P. Bingulac
University of Belgrade
Belgrade, Yugoslavia

P. J. West
University of Illinois
Urbana, Illinois

ABSTRACT

In late September 1982, Dr. S. P. Bingulac visited the University of Illinois and participated in the installation of a Computer-Aided Design (CAD) software package—called L-A-S, for Linear Algebra and Systems. Since then, various graduate students interested in control system design have used L-A-S as a software tool in their research work. This paper describes selected issues associated with the maintenance and enhancement of a large software CAD package. Also, we discuss some of our learning experiences with regard to making L-A-S easier to learn, easier to use, well-documented, "user-friendly", and numerically sound.

SUMMARY

There are many elements to successful software design. Laub [1] recently presented an excellent discussion of this topic. A good software design of a CAD package for control systems should try to encompass the needs of all while maintaining a measurable and meaningful standard of quality for the software. That is, fancy graphics and "user-friendliness" are important, however, the computational limitations of the algorithms being used to solve the problem are of significant concern as well.

L-A-S represents a modern example of a CAD package that has great potential for the control community and others interested in reliable solutions to linear system problems. L-A-S has been designed with several goals in mind. We enumerate a few of the ideas that characterize L-A-S:

(1) L-A-S is a control language. Essentially, L-A-S allows the user to take an algorithm that has been flowcharted using regular control system notation and immediately translate it into working computer code. The key concept of L-A-S is the notion of an operator. The user builds a program from the more than 80 linear algebra and control system operators.

(2) L-A-S consists of a hundred or more interconnected FORTRAN subroutines. Thus, L-A-S is very modular. This means that L-A-S can be easily adapted to perform on both large-scale computer installations (like our VAX) and small minicomputers (like a PDP-11 machine) where memory overlays are sometimes necessary.

(3) L-A-S is constructed from well-known and well-tested mathematical software packages, notably EISPACK [2] and LINPACK [3]. In general, it is safest to use these types of packages because their numerical stability has been extensively verified.

L-A-S is relatively easy to maintain. Of course, it is necessary to make the executable code (i.e., the machine language or binary) available to the general public. L-A-S stores files in the user's personal directory so the cost of having L-A-S on a computer system is just the cost of storing the L-A-S machine code.

The feedback from the various users of L-A-S has proved to be invaluable. The authors of L-A-S have constantly been modifying the existing software and adding new features based mainly upon comments from the users. Since

L-A-S is really a language interpreter, it is not surprising that our major learning experiences have been in the area of language improvement. This means that our understanding of how control engineers need and want to interact with computers is growing.

The latest additions to L-A-S include the so-called Multiple Operations Feature (MOF) and the capability for users to define their own L-A-S subroutines. The MOF allows the user to combine the effects of several operators on one line of an L-A-S program. In this way, L-A-S programs can be written in a very compact format. The idea of a subroutine is fundamental to any computer language. Hence, it was virtually inevitable that L-A-S would acquire its own subroutine facility.

In conclusion, we would like to state that CAD of control systems is still a young field of research. The software tools that we seek must address a wide variety of problems. Furthermore, we should pay particular attention to the computational limitations of the algorithms being used. Beyond this, there are several papers listed in the references that the interested reader may consult.

REFERENCES

[1] A. J. Laub, "Numerical Linear Algebra and Examples in Control," Proc. of 1983 ACC, San Francisco, California, 1983.

[2] B. S. Garbow, et al., Matrix Eigensystem Routines - EISPACK Guide Extension (Lecture Notes in Comp. Sci.), vol. 51, New York: Springer-Verlag, 1977.

[3] J. J. Dongarra, et al., LINPACK User's Guide, Philadelphia, Pennsylvania, SIAM, 1979.

[4] S. P. Bingulac and N. Gluhajic, "Computer Aided Design of Control Systems on Mini Computers using the L-A-S Language," Proc. of 1982 Symposium on Computer Aided Design of Multivariable Technological Systems, Purdue University, Indiana, 1982.

[5] S. Bingulac and J. Medanić, "An Efficient Pole Placement by Output Feedback," Proc. of 1982 Allerton Conference, Monticello, Illinois, 1982.

[6] W. F. Arnold and A. J. Laub, "RICPACK: Algorithms and Software for Matrix Riccati Equation," Proc. of IEEE Control Systems Society Symp. on Computer-Aided Control System Design, MIT, Cambridge, Massachusetts, September 1983.

SYSTOLIC ARRAYS FOR SOLVING LYAPUNOV AND RICCATI EQUATIONS

C. BOZZO, M. FOUQUES, E. SIFFREDI
CSEE-CETIA
La Valette du Var (France)

1. INTRODUCTION

Since it has been introduced, KALMAN filtering had many applications, especially for target tracking and trajectory estimation [4]. The computation of the KALMAN gain needs the solution of a RICCATI matricial equation ($P = \phi P \phi^T + \phi P H^T \left[H P H^T + R\right]^{-1} H P \phi^T + Q$), and many recursive algorithms have been developped which ensure,as the case may be,stability, precision or fastness. More, some algorithms need the solution of a LYAPUNOV matricial equation ($P = \phi P \phi^T + Q$).

Introduced by H.T. KUNG since 1978, systolic arrays have many applications in matrix computations [1], [2], [3]. They allow a large flow of operations with regular interconnections between elementary cells. That's why they are very well suited for VLSI and high speed computations.

The main systolic arrays that have been studied are :

. matrix multiplication
. matrix-vector multiplication
. LU-decomposition
. triangular linear systems
. triangular matrix inversion

Those are the basic arrays and other matricial computations can be done by a suitable combination of the basic arrays. For instance, the matrix inversion can be done by the following combination :

. LU-decomposition
. triangular matrix inversion
. matrix multiplication

So, the LYAPUNOV and RICCATI matricial equations can be solved by a combination of some of the basic arrays. However, the serial or parallel arrangement of basic arrays is optimal neither in number of cycles nor in number of elementary cells. So, it is better to reconsider the equations by developping some terms to obtain smaller arrays.

2. SYSTOLIC ARRAYS FOR LYAPUNOV EQUATION

2.1. Transient state

LYAPUNOV equation can be written in the following form :

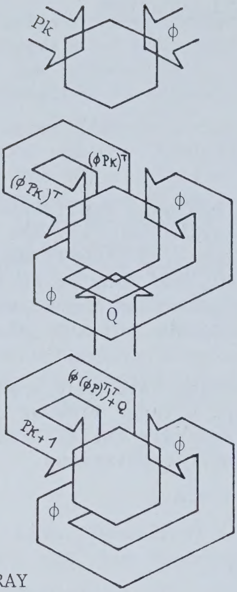
$$P_{k+1} = \left[\phi_{k,k+1} \cdot P_k + 0\right] \cdot \phi^T_{k,k+1} + Q_k$$

Then, the systolic array can be made with :

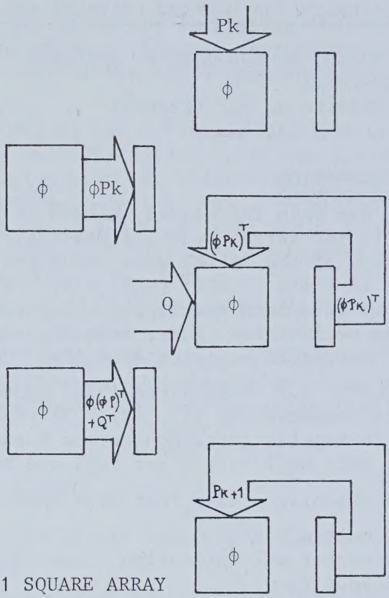
. 2 basic arrays, each array being used one time
. 1 basic array , used twice

The basic array S = A . B + C can be made with :

. an hexagonal structure
. a square structure



1 HEX. ARRAY



1 SQUARE ARRAY

2.2. Steady state

If it is obtained by iterations on the transient state, the previous arrays can be used.

If it is obtained by solving a $\frac{n\ (n+1)}{2}$ linear system, a combination of a LU decomposition systolic array and a triangular linear system systolic array will perform the final systolic array.

Note that the iterative solution is slower but do not need so many elementary cells.

3. SYSTOLIC ARRAYS FOR RICCATI EQUATION

The equation is considered in its recursive form, in the general case (n, m) , the array being made by a combination of basic arrays.

REFERENCES

[1] H.T. KUNG
Why Systolic Architectures ?
IEEE Computer, Jan 82, 37-46

[2] C.E. LEISERSON
Area-Efficient VLSI Computation
Dpt of Computer Science - CMU - october 81

[3] W.M. GENTLEMAN, H.T. KUNG
Matrix Triangulation by Systolic Arrays
SPIE, Vol 298, Real Time Signal Processing IV 81

[4] C. BOZZO
Le Filtrage Optimal & ses Applications aux Problèmes de Poursuite
Technique et Documentation, diffusion LAVOISIER - Paris 1983

OPTIMAL ALGORITHMS FOR COMPUTING DEPTHS AND LAYERS

BERNARD CHAZELLE
Computer Science Department
Brown University
Providence, RI 02912

Abstract

Let S be a set of n points in the plane. The *convex layers* of S is the set of convex polygons obtained by iterating on the following process: compute the convex hull of S and remove its vertices from S. This paper describes an optimal $O(n)$ space, $O(n \log n)$ time algorithm for computing the convex layers of S. Our algorithm outperforms the best method previously known by a factor of $\log n$ time. We also address the problem of determining the *depth* of a test point within the convex layers of S, i.e. the number of layers that enclose the test point. This is essentially a planar point location problem, for which we are able to give a very simple optimal solution.

1. Introduction

Let $S = \{p_0 \ldots, p_{n-1}\}$ be a set of n points in the plane. The *convex layers* of S, denoted $C(S)$ in the following, is the set of convex polygons defined iteratively as follows: compute the convex hull of S and remove its vertices from S (Fig.1). The convex layers of a point set can be seen as a natural extension of convex hulls. In [SH] Shamos mentions applications of this concept to pattern recognition and statistics: A problem central in *robust estimation* is that of evaluating an unbiased estimator that is not too sensitive to *outliers*, i.e. observations lying abnormally far from most others. To tackle the two-dimensional version of this problem, Tukey has suggested removing the outliers of a point set by *peeling* or *shelling* the set in the manner described above, iterating on this process until only a prescribed fraction of the original points remain [H]. Another illustration of the importance of convex layers was given in [CGL], where this notion was proven powerful enough to greatly improve upon the best algorithms previously known for the *halfplane range query problem*.

Finally, besides its practical relevance, the problem of computing the convex layers of a set of points is interesting in its own right, for it intuitively represents a geometric "equivalent" to sorting. Indeed, considering the various algorithms known for computing the convex hull of a set of points, one is tempted to draw a parallel with sorting algorithms. The Jarvis march [J] resembles selection sort, the standard divide-and-conquer convex hull algorithms [BS] smack of merge sort, and Eddy's algorithm [E] is strongly reminiscent of quicksort. There is, however, a fundamental difference which makes computing convex hulls often easier than sorting; the fact that the output is a convex polygon that may contain only a small fraction of the original points. This is what allows the existence of linear-expected-time algorithms for computing convex hulls under certain distributions of the points [AT,BS,T]. Knowing that similar results are provably impossible to obtain in the case of sorting [AHU], we can measure the intrinsic difference between the two problems (aside from the fact that convex hulls cannot be computed with linear decision trees [A]). One way of bridging this complexity gap is precisely to require the explicit computation of all the convex layers of the set of points, for it then becomes impossible to take advantage of the possible scarcity of the output in order to bound the time complexity of the problem.

This paper describes an $O(n)$ space, $O(n \log n)$ time algorithm for computing the convex layers of S. Computing the convex layers of n points requires $\Omega(n \log n)$ time [SH,Y], therefore our algorithm is optimal. The most efficient method previously known for this problem requires $O(n \log^2 n)$ time [OV]. We also address the problem of determining the *depth* of a test point within the convex layers of S, i.e. the number of layers that enclose the test point. This is essentially a planar point location problem, for which, therefore, optimal solutions already exist. We show, however, that the specific structure of the problem at hand allows us to use a new point location technique that is both optimal and practical, and avoids many of the complications of previous algorithms.

This paper is organized as follows: in Section 2, we introduce the basic ingredients of our method and prove a number of preliminary results. In Section 3, we give a description of the algorithm for computing convex layers, and in Section 4 we attack the problem of determining the depth of a test point.

2. The Ingredients

As observed by Overmars and van Leeuwen in [OV], each convex boundary in $C(S)$ can be represented as the concatenation of two convex polygonal lines, which we call *upper* and *lower* chains. The upper [resp. lower] chain consists of all the edges of the boundary oriented left-to-right [resp. right-to-left], clockwise. From now on, we will concentrate on the computation of, say, the upper chains of $C(S)$; the other case being strictly similar. Assume, wlog, that $p_0, \ldots, p_{n-1}$ appear X-sorted in this order. Consider the complete binary tree, T, whose leaves are $p_0, \ldots, p_{n-1}$, from left to right. Let $S(v)$ be the set of points stored at the leaves of the subtree rooted at the node v of T, and let $U(v)$ be the upper chain of the convex hull of $S(v)$, also referred to as the *upper hull* of $S(v)$. The union of all the edges of $U(v)$, for all nodes v in T, forms a planar graph, G, that can easily be shown to be a free tree. G is called the *hull graph* of S (Fig.2). Similarly, we define G', the hull graph of S with respect to lower chains.

Let U [resp. V] be an upper chain with vertices $u_1, \ldots, u_m$ [resp. $v_1, \ldots, v_p$], in clockwise order. We assume that all the vertices of U have smaller X-coordinates than the vertices of V. The *tangent* to U and V is defined as the unique edge joining U and V in the upper chain of the convex hull of $U_1 \bigcup U_2$. It is easy to compute this edge in $O(m+p)$ time, using a standard "gift-wrapping" technique. We omit the details of this straightforward procedure, and turn directly to the computation of G. This graph will be represented by means of adjacency-lists. Each vertex-to-edge list $V(p)$ will consist of two sublists, $V_1(p)$ and $V_2(p)$, defined as follows: $V_1(p)$ [resp. $V_2(p)$] is an angularly sorted doubly-linked list containing the vertices adjacent to p with smaller [resp. greater] abscissa than p. We define the *top edge* of $V_1(p)$ [resp. $V_2(p)$] to be the edge with minimum [resp. maximum] slope in $V_1(p)$ [resp. $V_2(p)$]. In addition, each of these lists is assumed to be supplied with a pointer to its top edge.

We compute G by divide-and-conquer. Assume that we have already computed the hull graphs of both $\{p_0, \ldots, p_{\lfloor n/2 \rfloor}\}$ and $\{p_{\lfloor n/2 \rfloor+1}, \ldots, p_{n-1}\}$. All we have to do is determine the tangent of these two sets, using the algorithm described earlier. It is easy to determine the two upper chains on which the tangent computation is based, using the extra pointer supplied with each vertex-to-edge sublist, so we omit the details.

3. The Convex Layer Algorithm

The basic idea is to iterate on the following process: make a copy of the names of the vertices on the upper chain, beforehand; then proceed to delete each of them from G, in turn, reconfigurating the hull graph after each deletion. Consider now the graph G' defined with respect to lower hulls. We proceed similarly, i.e. delete the vertices of the lower hull of G'. Next, we must reflect the *cross-deletions* in G and G', that is, delete the lower [resp. upper] hull vertices from G [resp. G']. To distinguish them from cross-deletions, we call the first type of deletions, *direct-deletions*. One difficulty in these operations is that the deletion of a single vertex can cause a major upset in the hull graph, and its reconstruction can take on the order of n operations. The key feature of the algorithm, however, is to ensure that these upsets always average out to yield an $O(n \log n)$ worst-case running time.

1. *Direct-deletions*

Imagine that we are currently at an arbitrary stage in the process, and that we wish to delete a vertex p of the upper hull currently processed. Let $v_1, \ldots, v_l$ be the nodes encountered in T when traversing the path from p to the root. Every v_i corresponds to a tangent edge t_i joining two upper hulls. Note that p lies below (or on) exactly one of these two hulls. It may be the case that because of previous deletions one of these hulls is empty (i.e. the one not above p). We can make this case consistent with the others, however, by assuming a dummy vertical tangent t_i. Let $w_1, \ldots, w_k$ be the subsequence of $\{v_1, \ldots, v_l\}$ that corresponds to tangent edges with p as an endpoint. This subsequence can be itself partitioned into the two subsequences formed respectively by the tangent edges of the form qp and pq, from left to right. It is easy to see that the former [resp. latter] subsequence corresponds exactly to $V_1(p)$ [resp. $V_2(p)$].

The first step consists of retrieving from $V(p)$ the sequence of tangent edges $t_{i_1}, \ldots, t_{i_k}$ corresponding to the nodes $w_1, \ldots, w_k$. This involves *merging* the lists $V_1(p)$ and $V_2(p)$ according to the following criterion: q_1 precedes q_2 iff the node in T corresponding to the tangent pq_1 is a descendant of the node corresponding to pq_2. This merge is done very easily in $O(l)$ operations by traversing the path from v_1 to v_l in T. In practice, of course, we can emulate the tree T by observing that if $p = p_i$, the path to the root corresponds to the left-to-right sequence of bits in the binary representation of the integer i. The bits of i are to be interpreted as follows: 0 and 1 correspond respectively to left and right turns down the tree. Therefore, the merge follows the prescription: q_1 precedes q_2

iff its index has a longer common prefix with i than the index of q_2 has. For the sake of consistency, we define t_0 as a dummy tangent extending vertically from p towards $y = -\infty$. t_0 will be considered indifferently of type pq or qp, depending on the situation at hand. This addition is meant to ensure that p always has at least one tangent of both types.

For any node in T, define $G(v)$ as the subgraph of G induced by the leaves of the subtree of T rooted at v. Deleting p from G involves updating the sequence of graphs $G(w_1), \ldots, G(w_k)$, in this order. Suppose that we are about to update $G(w_i)$. Wlog we can assume that the corresponding tangent t is of the form pq. Let ap [resp. pb] be the last tangent of type qp [resp. pq] that we processed. Assume for the time being that both a and b are well-defined vertices. Since $G(w_{i-1})$ has been already computed, its upper hull is now available, and as may be noticed, has its portion between a and b lying below the wedge (pa, pb). Let a', b', and c' be the vertices of the hulls currently examined, following a in clockwise order, b in counterclockwise order, and c in counterclockwise order (Fig.3). We next define the angles $\alpha = (aa', ap)$, $\beta = (bp, bb')$, $\gamma = (cp, cc')$, $\delta = (pc, ap)$, and $\epsilon = (pb, pc)$. Updating $G(w_i)$ involves "pulling down" the vertex p until it disappears from the upper hull of the points in $G(w_i)$. To do so, we must compute which of the five angles $\alpha, \beta, \gamma, \delta, \epsilon$ is the first to become null.

We can avoid angle calculations altogether if we simply compute the intersection of the vertical line passing through p with the lines passing through aa', bb', and cc' (Fig.4). The vertex a', b', or c' that determines the next location of p is said to be *wrapped*. We iterate on this process until δ or ϵ becomes null, whichever happens first. Figure 5-A,B illustrates the two important cases. Note that the addition of the dummy tangent t_0 allows us to treat the case where any of a, b, or c is not properly defined, in similar (yet simpler) ways. We may omit the details of the special cases, which are quite straightforward to handle. In general, we can distinguish among three kinds of wrapping: A point wrapped by a subsegment of ap, pb, or pc is said to be *lower-wrapped, inner-wrapped*, or *upper-wrapped*, respectively.

It is important to observe that since p is on the upper hull of G, all the edges of the form pa, pb, and pc are top edges in $V_2(a)$, $V_1(b)$, and $V_1(c)$, respectively. This allows us to gain access to the vertices a', b', c' as well as update the lists $V_2(a)$, $V_1(b)$, and $V_1(c)$ in constant time. This remark leads us to the following result.

Lemma 1. The number of operations involved in updating $G(w_i)$ is proportional to the number of vertices wrapped during the updating.

We will successively prove the correctness of the algorithm and study its complexity.

Lemma 2. The algorithm outlined above correctly computes the new shape of G after the direct-deletions of all the vertices on its upper hull.

Proof: Let $C(v)$ be the upper hull of the vertices in $G(v)$. The first observation to make is that removing p from G causes the disappearance of all the edges in $V(p)$, but only these. Furthermore, it causes the introduction of a number of new edges that will all be in $C(v_1), \ldots, C(v_l)$ after the deletion of p. We will show by induction that the algorithm properly updates each $C(v_i)$. Let v_{i-1} and v be the two sons of v_i. We must distinguish between two cases:

1. p is not an endpoint of the tangent associated with v_i. $C(v_i)$ is still formed by joining $C(v)$ and $C(v_{i-1})$ with the same tangent associated with v_i as before. By induction, $C(v_{i-1})$ will have then be properly updated, so no additional work is required.
2. The tangent associated with v_j is of the form pq (Fig.3) (the case qp is strictly similar). The procedure given above correctly computes the new upper hull of the vertices of $G(v) \bigcup G(v_{i-1})$, provided that the angles $\alpha, \beta, \gamma, \delta, \epsilon$ in Figure 3 are well defined. γ, δ, and ϵ are already defined in the original setting of G; as for α and β, they originate from the fact that the new setting of $C(v_{i-1})$ differs from the previous one only between a and b, and that the part of $C(v_{i-1})$ between these two vertices lies strictly below the wedge (pa, pb). ▮

2. *Cross-deletions*

We can now look at the cross-deletion process: the vertices to be deleted are on the lower chain of the convex hull of S. We remove them one at a time, say, in clockwise order, applying the method described previously. A few remarks are in order, however. Let p be the vertex to be removed. By construction, it is obvious that the two top edges in $V(p)$, i.e. pa in $V_1(p)$ and pb in $V_2(p)$ always form a convex angle, i.e. $(pa, pb) \leq \pi$. Since on the other hand, p is on the convex hull of S, it follows that one of the lists $V_1(p)$ or $V_2(p)$ has to be empty (i.e. reduced to the dummy edge t_0). Note that this remains true for all the lower hull vertices yet to be removed, even after any number of them have been already cross-deleted. This shows that cross-deletions correspond to one of the special (easy) cases mentioned earlier.

The second observation has to do with the access to the adjacent vertices of p. Suppose wlog that $V(p) = V_1(p) = \{pa_1, \ldots, pa_u\}$, with pa_1 being the top edge. It is crucial that we be able to gain constant-time access to the location where p is stored in each of the lists $V_2(a_1), \ldots, V_2(a_u)$. As we have seen it earlier, this is indeed a requirement if we wish to preserve the result of Lemma 1, i.e. if we want to ensure that the time to cross-delete p is proportional to the number of wrappings. Unfortunately, it is no longer true that every edge pa_i will be a top edge in $V_2(a_i)$, for p is not on the upper hull any more. The next result suggests a way to circumvent this difficulty.

Lemma 3. At most one edge pa_i $(i = 1, \ldots, u)$ is not a top edge in $V_2(i)$.

Proof: Consider two adjacent edges vw and vz in G. We say that vz *dominates* vw if there exists a vertical line that intersects vw and vz in two distinct points, w' and z', respectively, with the Y-coordinate of z' dominating the Y-coordinate of w'. Let wh be another edge adjacent to vw. It is easy to see that, by construction of G, the edges vz and wh can never both dominate vw. Indeed, both vz and wh completely cover vw from above, i.e. they lie above vw and each of their projections on the X-axis contains the projection of vw. This shows that if vz and wh both dominated vw they would cross, which is impossible. This immediately implies that all the edges $pa_2, \ldots, pa_u$ are top edges in their respective list $V_2(a_2), \ldots, V_2(a_u)$. ∎

Since pa_1 is the only edge that does not lend itself to direct access, we can afford to scan the list $V_2(a_1)$ linearly, for it is easy to show that it contains at most $\log n$ entries. This will add a term $O(\log n)$ in the time to cross-delete p, which will be of no consequence since we are aiming for an $O(n \log n)$ time algorithm for computing all the layers. We are now ready to examine the complexity of the overall algorithm.

Theorem 1. It is possible to compute the convex layers of a set of n points in $O(n \log n)$ time, using $O(n)$ space.

Proof: Let's look first at the cost of direct-deletions. In the course of updating $G(v_i)$, for $i = 1, \ldots, k$, any vertex either lower- or upper-wrapped is said to be *promoted,* while any vertex inner-wrapped is said to be *confirmed.* The crucial observation is that, in the course of the whole computation of $C(S)$,

1. The number of operations is $O(n \log n +$ total number of promotions and confirmations).
2. Vertices are never confirmed twice in a row, i.e. if they are confirmed more than once, they must be promoted at least once in between each confirmation.

The first point follows directly from previous remarks. To understand the second feature of promotion, let's take a closer look at the wrapping of a point p_u. Let w be the highest-level node of T (immediately before the wrapping) such that p_u is a vertex of $C(w)$. If p_u is lower-wrapped or upper-wrapped, it immediately becomes a vertex of an upper hull $C(v)$ that lies above $C(w)$. It then follows that v is strictly higher than w in T, and replaces w as the the highest-level node z of T such that p_u is a vertex of $C(z)$. Since T has height $\log n$, each point can be thus promoted at most $\log n$ times.

Let's turn next to the *confirmation* process. In some sense, pulling down ap and bp is a replay of previous action that is only meant to guide the pulling of ap and pc. More precisely, consider any vertex p_u that is inner-wrapped. It is easy to see that it was last lower- or upper-wrapped, therefore this wrapping corresponded to a promotion, which proves our claim. We have thus shown that a point cannot be wrapped more than $2 \log n$ times, which shows that the overall time cost of direct-deletions is $O(n \log n)$.

We can treat cross-deletions in an identical way, charging each vertex wrapped in the manner just described. Each deleted vertex may be also charged an extra $O(\log n)$ to account for the linear scan of one of the lists $V_1(*)$ or $V_2(*)$. This completes the proof. ∎

4. Computing the Depth of a Point

The *depth* of a query point q within $C(S)$ is defined as the number of convex layers that enclose q —see [OV,SH] for practical applications of this notion. Let $S_1, \ldots, S_k$ be the convex layers of S, numbered inwards. Rather than computing only the depth of q, let's consider the more general problem of determining the pair of consecutive layers (S_m, S_{m+1}) between which q lies. This pair may be reduced to a single layer if q lies either outside S_1 or inside S_k. The index m will be denoted $m(q)$ from now on, with the convention that $m(q) = 0$ if q lies outside of S_1.

This problem is essentially a *planar point location problem,* i.e. a problem of determining which region of a straight-line subdivision of the plane contains q. Two optimal algorithms are known for solving this problem [K,LT], as well as a near-optimal method [P,BI], more practical than the optimal ones. We wish to show here how planar point location in the graph $C(S)$ can be done optimally, combining the simplicity of [P,BI] with the performance of [LT,K]. More precisely, we will present a practical method for computing $m(q)$ in $O(\log n)$ time, using $O(n)$ storage. We describe the algorithm in two steps: to begin with, we describe an $O(n \log n)$ space, $O(\log n)$ time method, to which we then apply the concept of *filtering search* introduced in [C] so as to reduce the space requirement to $O(n)$.

1. *The Preliminary Algorithm*

Let's organize the set of layers in $C(S)$ into a complete binary tree T, such that each node is associated with a layer of $C(S)$, and an inorder traversal of T corresponds to the order $S_1, \ldots, S_k$. Each node v in T contains a pointer to a data structure, $DS(v)$, built from the layer associated with v, which we denote $S(v)$. In a first stage, we may suppose that $DS(v)$ is simply an array giving a clockwise description of $S(v)$. Let C be a point inside S_k and let $R(q)$ be the ray emanating from C that passes through the point q. Since C also lies inside the convex polygon $S(v)$, we can use the array $DS(v)$ to compute by binary search the intersection of $S(v)$ and $R(q)$ [SH]. This allows us to determine in $O(\log n)$ time whether q lies inside or outside of $S(v)$.

Setting the tree T in this fashion leads directly to an $O(n)$ space, $O(\log^2 n)$ time algorithm, which we can somewhat speed up, at the expense of a factor of $\log n$ space. To do so, we apply a standard technique [W], which involves adding extra pointers between adjacent nodes of the search tree. Let $R(v)$ be the (angularly) sorted list of rays $R(p)$, for all vertices p of $S(v)$. We next define $L(v)$ as the sorted list of rays in $R(v) \bigcup D(v)$, with $D(v) = [\bigcup R(z) \mid \text{all descendants } z \text{ of } v]$. We can compute all the lists $L(v)$ recursively, in $O(n \log k) = O(n \log n)$ time, simply merging $R(v)$, $L(v_1)$, and $L(v_2)$, whereby v_1 and v_2 are the sons of v.

The next step is to refine the boundary of $S(v)$ with the added information of $L(v)$. More precisely, we augment the set of vertices of $S(v)$ by adding to it all the intersections between $S(v)$ and the rays of $L(v)$. This creates a new boundary, $S^*(v)$, which is simply a "copy" of $S(v)$ supplied with additional vertices. Note that any edge in $S^*(v)$ forms a wedge with respect to C that is *covered* by exactly one wedge from $S^*(v_1)$ and one wedge from $S^*(v_2)$. This allows us to define two pointers from each edge of $S^*(v)$, one to the two edges in $S^*(v_1)$ and $S^*(v_2)$ that correspond to a covering wedge. The data structure $DS(v)$ can now be defined as a clockwise description of $S^*(v)$ along with this set of extra pointers. It is easy to see that the overall structure can be computed in $O(n \log n)$ time.

The point location algorithm will now proceed as follows: let r be the root of T. Use a binary search to determine, in $O(\log n)$ operations, the edge e of $S^*(r)$ intersected by the ray $R(q)$. Testing the relative position of e and q will indicate whether we should iterate the search in the left or right subtree. Once this has been determined, we can follow one of the extra pointers associated with e, so as to avoid any further binary search. This will, indeed, lead us directly to the new edge to be compared against q. This new algorithm will thus yield an $O(\log n)$ search time, but at the price of added storage, i.e. $O(n \log k) = O(n \log n)$.

2. *The Optimal Algorithm*

Lemma 4. It is possible to find, in $O(n)$ time, a sequence V of convex layers among $\{S_1, \ldots, S_k\}$, such that the total number of vertices in V is $O(n/\log n)$ and at most $4 \log n$ layers can be found between any pair of consecutive layers in V.

Proof: Omitted. ▮

We are now in a position to apply the previous algorithm for computing $m(q)$ in order to report the pair of consecutive layers in V that enclose a test point. The reduced size of V will ensure that only $O(\frac{n}{\log n} \log(\frac{n}{\log n})) = O(n)$ preprocessing is now needed, instead of $O(n \log n)$. At this stage, the problem is to locate q among $O(\log n)$ convex layers. This can be done by computing the intersection of each of these layers with the ray $R(q)$, using the hive graph structure of [C]. This is a simple data structure that allows us to solve the following problem optimally: report which of m given disjoint segments intersect a query segment that has a fixed slope. The algorithm requires $O(m)$ storage and gives a response-time $O(k + \log m)$, where k is the number of segments to report. This allows us to perform the computation needed in our case in $O(\log n)$ time. We can conclude with the main result of this section.

Theorem 2. It is possible to locate a point among the convex layers of n points, hence also compute its depth, in $O(\log n)$ time, using $O(n)$ storage.

5. Conclusions

The main contribution of this work has been to present an optimal algorithm for computing the convex layers of a point set. We also provided a new method for locating any test point within the set of layers, that is both practical and optimal.

REFERENCES

[A] Avis, D. *Comments on a lower bound for convex hull determination*, ILP 11(3), Nov. 1980.

[AHU] Aho, A.V., Hopcroft, J.E. and Ullman, J.D. *The design and analysis of computer algorithms*, Addison-Wesley, 1974.

[AT] Akl, S.G., Toussaint, G.T. *Efficient convex hull algorithms for pattern recognition applications*, Proc. 4th Int. Joint Conf. on Pattern Recognition, pp. 483–487, Nov. 1978.

[BS] Bentley, J.L., Shamos, M.I. *Divide-and-conquer for linear expected time*, Info. Proc. Lett., 7(1978), pp. 87-91.

[BI] Bilardi, G. *Average case analysis of an adjacency map searching technique*, Tech. Report R-928, UILU-ENG 81-2259, University of Illinois, Dec. 1981.

[C] Chazelle, B. *Filtering search: A new approach to query-answering*, Proc. 24th FOCS Symp., Tucson, 1983.

[CGL] Chazelle, B., Guibas, L., Lee, D.T. *The power of duality*, Proc. 24th FOCS Symp., Tucson, 1983.

[E] Eddy, W.F. *A new convex hull algorithm for planar sets*, ACM Trans. Math. Software, Vol. 3 (1977), pp. 398-403 and pp. 411-412.

[H] Huber, P.J. *Robust statistics: a review*, Ann. Math. Stat. 43(3), pp. 1041–1067, 1972.

[J] Jarvis, R.A. *On the identification of the convex hull of a finite set of points in the plane*, Info. Proc. Lett. 2, 1973.

[K] Kirkpatrick, D.G. *Optimal search in planar subdivisions*, SIAM J. on Comp., Vol. 12, No. 1, February 1983.

[LT] Lipton, R.J., Tarjan, R.E. *Applications of a planar separator theorem*, SIAM J. Comp., 9 (3), 1980.

[OV] Overmars, M.H., van Leeuwen, J. *Maintenance of configurations in the plane*, Journal of Computer and System Sciences, 23, p. 166-204, 1981.

[P] Preparata, F.P. *A new approach to planar point location*, SIAM J. Comput., Vol. 10 (3), Aug. 1981.

[SH] Shamos, M.I. *Computational geometry*, PhD thesis, Yale University, 1978.

[T] Toussaint, G.T. *Pattern recognition and geometrical complexity*, Proc. 5th. Int. Conf. on Pattern Recognition, Dec. 1980.

[W] Willard, D.E. *New data structures for orthogonal queries*, To appear in SIAM J. Comp.

[Y] Yao, A.C. *A lower bound to finding convex hulls*, Journal of the ACM, 28(1981), pp. 780-789.

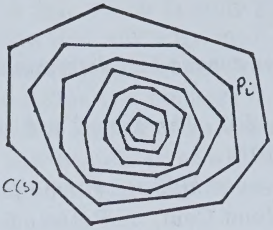


Figure 1

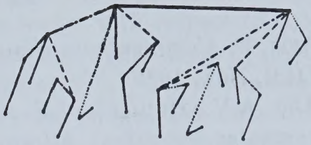


Figure 2

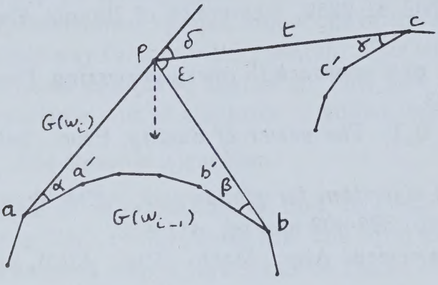

p
δ
t
c
γ
ε
c′
G(w_i)
a′
b′
α
β
a
G(w_{i-1})
b

Figure 3

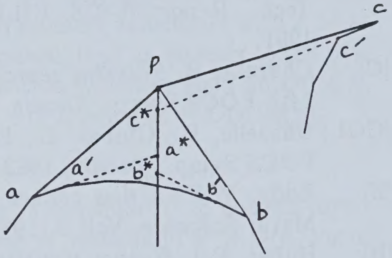

c
p
c′
c*
a*
a′
b*
b′
a
b

Figure 4

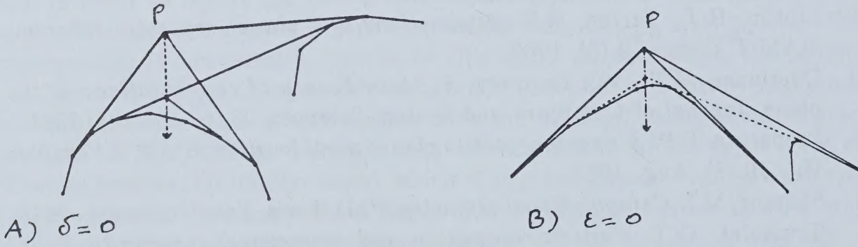


Figure 5

A SIMPLE HIDDEN-LINE ALGORITHM FOR RECTILINEAR POLYGONS

JÖRG-RÜDIGER SACK
School of Computer Science
McGill University
805 Sherbrooke St. West
Montreal, Quebec H3A 2K6
Canada

ABSTRACT

In this paper we present a simple, one stack, backtracking algorithm to solve the hidden-line problem for rectilinear polygons. The results are based on a new tool in computational geometry, a labeling scheme of the polygonal edges.

1. INTRODUCTION

To render a realistic picture of an object displayed on a graphics device, we have to decide which parts of the object are visible and which parts are invisible. The problem is of fundamental importance in computer graphics and thus a variety of hidden-line removal algorithms have been designed. We assume that the objects to be displayed are given as polygons. A polygon is simple if no two non-consecutive edges intersect.

A linear time algorithm to solve the hidden line problem for simple polygons has been presented in [Ell], but their algorithm is rather involved and utilizes three stacks. A special hidden-line problem, determining the x-y hull of a rectilinear polygon, has been studied in [Mon]. In this paper we present a very simple, linear-time algorithm to solve the hidden-line problem for simple rectilinear polygons.

The construction of the convex hull of a simple polygon or a set of points is one of the most extensively studied problems in computational geometry. Very simple, linear-time convex hull-algorithms for polygons with specific shape characteristics have been designed [Tou]. One example is Sklansky's convex hull-algorithm which efficiently constructs the convex hull of a weakly externally visible polygon [Skl]. Linear-time convex hull-algorithms for simple polygons, however, tend to be more complicated in that these algorithms use more than one stack, or their program structure is complex [McC, Lee]. As we know from the history of convex hull algorithms, complex program structures may sometimes result in difficult proofs of correctness, if not even faulty algorithms. In combining the hidden-line algorithm presented here with Sklansky's algorithm, we obtain a very simple and efficient convex hull-algorithm for rectilinear polygons.

2. LABELING SCHEME

Our solution to the hidden-line problem for simple, rectilinear polygons is based on a new tool characterizing the polygon's winding properties. We will first describe this tool, which is a labeling scheme of the edges. Then we will prove several results on the relationship between visibility in polygons and the labeling scheme.

Labeling scheme: Let $P = \{p_1, \ldots, p_n\}$ be an n-vertex polygon and let ϕ_i denote the exterior angle at p_i (see Figure 1). Let $e_1, \ldots, e_n$ denote the edges of P, with $e_i = (p_{i-1}, p_i)$. As usual, operations on indices are taken modulo n. Furthermore let e_1 be the edge with largest y-value, where ties are broken in favor of the rightmost edge. We will now define the labeling scheme:

$\text{label}(e_1) := 0$ (* the first edge is arbitrarily labeled 0 *)

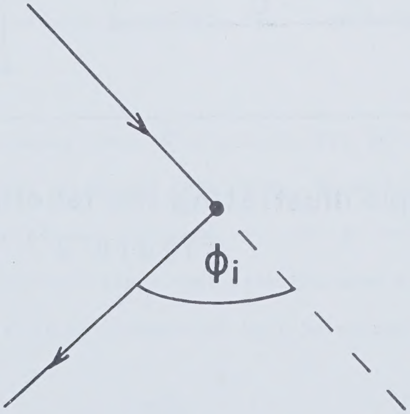
$\text{label}(e_i) := \text{label}(e_{i-1}) + \phi_i$

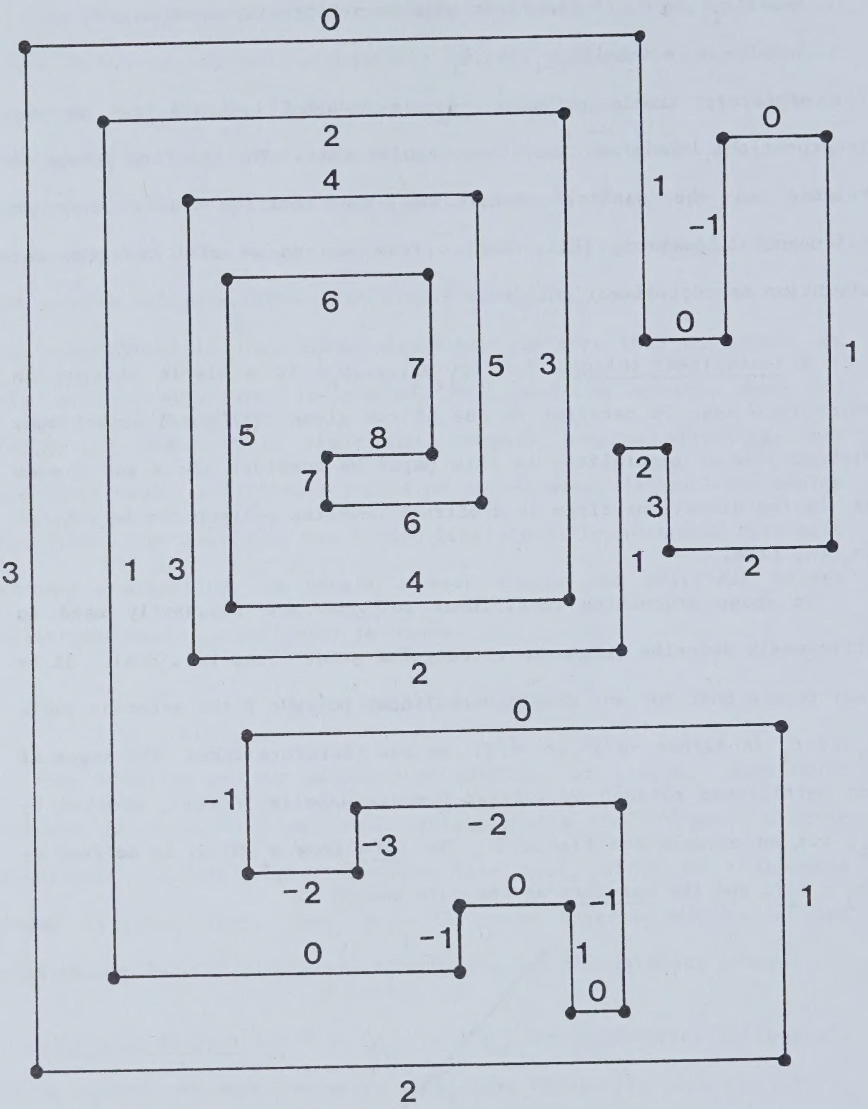
For arbitrary simple polygons, ϕ_i is bounded by $-\pi<\phi_i<\pi$. We can interpret the labels as cumulative angular sums. The labeling scheme is related to the winding number and the rotation index know in differential geometry [Kli, Car]. From now on we will restrict our attention to rectilinear polygons.

A rectilinear polygon $P = \{p_1, p_2, \ldots, p_n\}$ is a simple polygon in which each edge is parallel to one of two given orthogonal directions. Without loss of generality, in this paper we consider the x and y axes as the two directions since an arbitrary n-vertex polygon can be rotated in O(n) time.

In image processing rectilinear polygons are frequently used to efficiently describe images on rectangular grids [Fen, Pav, Sck]. It is easy to see that for any simple rectilinear polygon P the exterior angle ϕ_i at p_i is either $+\pi/2$,or $-\pi/2$. We can therefore label the edges of any rectilinear polygon by integer numbers $\text{label}(e_i)/(\pi/2)$, denoted by l_i, for an example see Figure 2. The turn from e_i to e_j is defined as $|l_j - l_i|$, and the net-turn as the turn modulo 4.

Figure 1




0
2
4
6
7
8
7
6
5
5
4
3
3
1
3
2
1
1
0
-1
0
1
2
3
1
2
0
-1
-2
-3
-2
0
-1
-1
1
0
0
1
2

An example illustrating the labeling-scheme

Figure 2

3. PROPERTIES OF THE LABELING SCHEME

In characterizing the visibility of objects, we can either adopt a perspective model or a parallel model [Yao]. In the perspective model we assume that a point q is located in the plane, and a solution to the hidden-line problem consists of finding all points p in P that are visible from q. In the parallel model, we assume that rays emanate from a source, a point at infinity and form a fixed angle θ with the y-axis. In this model a point p is visible if there exists a point q at infinity s.t. the ray qp with angle θ does not intersect P. We then say that p is visible from ∞. For pedagogical reasons we are always assuming that our visibility point, ∞, is to the right of P, and that the rays $(r*\cos(\theta),\sin(\theta))$, for $r \to \infty$ are parallel to the x-axis, i.e. $\theta=0$. The algorithm presented here can easily be modified to encompass all possible values of θ. As horizontal edges can be reconstructed from the subset of all vertical edges, it is sufficient to examine vertical edges only. We will now show that depending on the viewpoint only edges with specific labels are visible.

Lemma 1: Labels of horizontal edges are even and labels of vertical edges are odd.

Lemma 2: If a vertical edge is visible from a point at ∞ then its label is 1.

We could obtain analogous results for visibility points located at infinity above P, below P, or to the left of P. We define a horizontal cut, h through a rectilinear polygon P, as a horizontal line intersecting all those vertical edges that have the same y-coordinate as h. Let the list of all k edges intersected by h be sorted, where e_{i_1} is the leftmost edge and e_{i_k} the rightmost edge. The corresponding label-list, denoted by $(l_{i_1}, \ldots, l_{i_k})$, is called a horizontal cut-sequence of length k. Similarly we could define vertical cut-sequences. The following Theorem will be helpful in understanding the correctness of our algorithm.

Theorem 1: Let $(l_{i_1}, \ldots, l_{i_k})$ be a horizontal cut-sequence of length k, then

(i) $l_{i_1} = 3$, $l_{i_k} = 1$, and (ii) $|l_{i_j} - l_{i_{j+1}}| = 2$.

4. THE ALGORITHM

We will now solve the hidden-line problem for rectilinear polygons P. Recall that p_1 is the rightmost vertex with maximum y-coordinate. Let $C_{1,k}$ be the polygonal chain from p_1 to the rightmost vertex with lowest y-value, called p_k. Between the vertices p_k and p_1 there exist two simple chains: the right chain $RC(p_1,p_k)$ and the left chain $LC(p_k,p_1)$. We will show how to solve the hidden-line problem from a point located at $(+\infty, 0)$, examining therefore the right chain $RC(p_1,p_k)$ only.

For ease of description assume that the edges e_i of P are already labeled and that l_i are the corresponding labels. We maintain a stack, S, containing the visibility chain for all edges so far considered. The points in S are stored in order of decreasing y-coordinates, where the lowest point is at the top of the stack, referred to as TOP. We will say that a point p is to the right of a point q, if the x-coordinate of p is greater than the x-coordinate of p. Similarly we use to the left of, higher than, and lower than. Let a,b be two points then we will denote by $a\#b$ the point whose x-coordinate is that of a and whose y-coordinate is that of b.

```
Algorithm: hidden-line problem for rectilinear polygons
Input: the polygonal chain $C_{1,k}$
Output: the visibility chain from a point at infinity
PUSH( $p_1$ ); (* stack initialization *)
PUSH( $p_2$ );
For i := 2 to k do
  if $l_i$ = 1 and $p_i$ is lower than TOP then
  begin
    PUSH( $p_i$ # TOP);
    PUSH( $p_i$ )   (* the visibility chain is augmented *)
  end
  else if $l_i$ = -1 and $p_{i-1}$ is not higher than TOP then
  begin
    while $p_i$ is to the right of TOP and above TOP do POP;
          (* shrink the visibility chain to the level of $p_i$ *)
    PUSH(TOP# $p_i$ )
  end.
```

Theorem 2: The algorithm solves the hidden-line problem for rectilinear polygons in linear time.

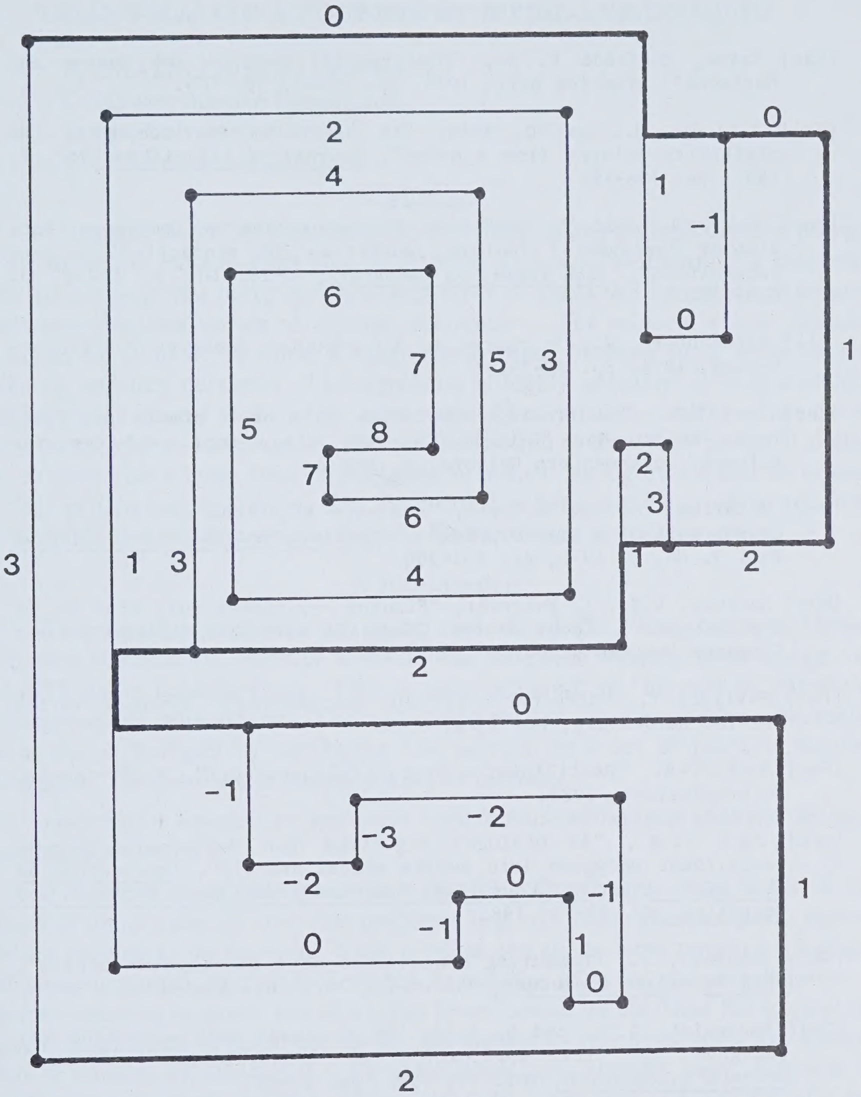
An example illustrating the algorithm presented here, is given in Figure 3, where bold lines represent the solution to the hidden-line problem from a point located at $(+\infty, 0)$.

If, in the parallel model, the rays are not parallel to the x-axis, i.e. the view-point is not located at $(+\infty, 0)$, then the visible edges have at most two distinct labels. We can determine these labels and modify the push and pop-operations accordingly. In [Sac], we have shown how to solve the hidden-line problem in the perspective model and in the case that the point of visibility is located inside the polygon. For this only minor modifications to the algorithm described above are necessary.

5. CONCLUSIONS

We have introduced a simple labeling scheme for edges of rectilinear polygons. The scheme is based on angular properties of polygons and carries some winding information. The predicate "visibility from ∞", is a global property of the polygonal structure and not a local angular property. The labeling scheme is a tool to extract such properties easily. A very simple, one stack, backtracking algorithm was presented to solve the hidden-line problem for rectilinear polygons. Using Sklansky′s idea, adding one additional if-statement would convert this algorithm into a very simple, one stack convex hull algorithm.

The labeling scheme itself can be applied to solve a variety of other interesting problems in computational geometry, robotics and graphics [Sac]: the perspective hidden-line problem, visibility in polygons, similarity between polygons, movements of robots, shortest paths problems, and several tests for polygon properties. The ideas described here for rectilinear polygons are extendible to arbitrary simple polygons.


0
2
4
6
7
5
3
8
7
6
4
3
1
3
5
0
1
-1
0
1
2
3
1
2
2
0
-1
-2
-3
-2
0
-1
-1
1
0
0
1
2

The hidden-line problem from a point at $(+\infty, 0)$

Figure 3

6. REFERENCES

[Car] Carmo, Manfredo P. do, "Differential Geometry of Curves and Surfaces", Prentice Hall, 1976, pp. 30-41, 390-404.

[Ell] ElGindy, H., and D. Avis, "An algorithm for computing the visibility polygon from a point", Journal of Algorithms, Vol. 2, 1981, pp. 186-197.

[Fen] Feng, H., and T. Pavlidis, "Decomposition of polygons into simpler components: feature generation for syntactical pattern recognition", IEEE Trans. on Computers, C-24, No. 6, 1975, pp. 636-650.

[Kli] Klingenberg, W., "A course in differential geometry", Springer-Verlag, 1978, pp. 21-29.

[Lee] Lee, D.T., "On finding the convex hull of a simple polygon", Tech. Report No. 80-03-FC-01, Dep. Elec. Engr. and Computer Science, Northwestern University, 1980.

[McC] McCallum, D., and D. Avis, "A linear algorithm for finding the convex hull of a simple polygon," Information Processing Letters, Vol. 9, No. 5, 1979, pp. 201-206.

[Mon] Montuno, D.Y., A. Fournier, "Finding x-y convex hull of a set of x-y polygons", Tech. Report CSRG-148, Dynamic Graphics Project Computer Systems Research Group, 1982.

[Pav] Pavlidis T., "Structural pattern recognition", Springer Verlag Berlin, Heidelberg, New York, 1977.

[Sac] Sack,J.-R., "Rectilinear computational geometry", Ph.D.-thesis, in preparation, 1983.

[Sck] Sack J.-R., "An O(nlogn) algorithm for decomposing simple rectilinear polygons into convex quadrilaterals", Proceedings of The 20th Allerton Conference on Communication, Control and Computing, pp. 64-74, 1982.

[Skl] Sklansky, J., "Measuring concavities on a rectangular mosaic", IEEE Trans. on Computers, Vol. C-21, 1972, pp. 1355-1364.

[Tou] Toussaint, G.T., and D. Avis, "On a convex hull algorithm for polygons and its application to triangulation problems", Pattern Recognition, Vol. 15, 1982, pp. 23-29.

[Yao] Yao, F.F., "On the priority approach to hidden-surface algorithms", Proc. 21th Annual Symposium of Computer Science, Oct. 1980.

TRIANGULATING A POLYGON BY DIVIDE-AND-CONQUER

B. CHAZELLE and J. INCERPI
Computer Science Department
Brown University
Providence, RI 02912

Abstract

This paper describes a new method for triangulating a simple n-sided polygon. The algorithm runs in time $O(n \log s)$, with $s \leq n$. The quantity s measures the ***sinuosity*** of the polygon, i.e. the number of times the boundary alternates between complete spirals of *opposite* orientation. The value of s is in practice a small constant or at worst a very slow-growing function of n, even for extremely winding polygons. The algorithm is highly adaptive; it uses a divide-and-conquer scheme that has the advantage of not requiring sorting or the use of any balanced tree structure. Aside from the notion of sinuosity, we are also able to characterize a large class of polygons for which the algorithm can be proven to run in $O(n \log \log n)$ time. The algorithm has been implemented, tested, and has shown to provide an advantageous alternative to previous methods.

1. Introduction

Triangulation problems are many and varied [T]. One common thread between them is the attempt to refine the notion of *neighborhood* among the objects under consideration. This is quite apparent in the case of polygonal triangulations, whereby one is asked to augment the set of boundary adjacencies of a simple polygon by expressing the polygon as a set of pairwise disjoint triangles. The problem is simply stated as follows:

Decompose a simple polygon into a set of non-overlapping triangles without adding new vertices.

The relevance of this question is two-fold. On the one hand it lies at the basis of many other geometrical problems (e.g. visibility, shortest-path, region-filling problems); on the other hand it raises one of the most puzzling questions of computational geometry: does the knowledge of a simple path between n points allow us to break the $\Omega(n \log n)$ lower bound on the time for computing any triangulation of these points [S]. No answer to this question is yet known, but a number of $O(n \log n)$ time algorithms have already been proposed to triangulate a simple n-sided polygon [C,G,H]. See also [TA,SV] for efficient solutions to variants of the problem. We propose in this paper to extend our current knowledge of the problem by describing a new algorithm which, in many aspects, outperforms all previous methods.

The aim of this work is to present an algorithm whose complexity reflects the *shape-complexity* of the polygon. We introduce the concept of *sinuosity* for this purpose. Roughly speaking, the sinuosity of a polygon, denoted s, measures the number of times the boundary alternates between complete spirals of *opposite* orientation. We always have $s \leq n$, but the value of s is in practice a small constant or at worst a very slow-growing function of n, even for extremely winding polygons. We describe a triangulating algorithm that runs in time $O(n \log s)$. The algorithm is highly adaptive; it uses a divide-and-conquer scheme that has the advantage of not requiring sorting or the use of any balanced tree structure. Aside from the notion of sinuosity, we are also able to characterize a large class of polygons for which the algorithm can be proven to run in $O(n \log\log n)$ time. The algorithm has been implemented, tested, and has shown to provide an advantageous alternative to previous methods.

2. Decomposing a Polygon into Trapezoids

2.1 The Basic Ingredients

The first and main goal of the algorithm is to compute the ***vertical decomposition*** of the polygon P, denoted $VD(P)$. This is the unique partition of P obtained by drawing a vertical line through each vertex of P, extending each line as long as it does not properly cross the boundary of the polygon (Fig.1). We will show later on that it is straightforward to derive a triangulation of P once its vertical decomposition is available. Note that for $VD(P)$ to be well-defined, P must be simple, but does not necessarily have to be closed. We can extend the notion of vertical decomposition to any simple, oriented polygonal line L. Before proceeding, we introduce some terminology: let t be a vertical segment intersecting L at some point p, and let D be a disk centered at p. If the radius is chosen small enough, we can always ensure that either L subdivides D into exactly two parts, provided that p is not an endpoint of L. Let C be the part with respect to which the orientation of L runs clockwise. If the intersection of t with D lies completely in C, or if p happens to be an endpoint of L, we say that t is in ***right-contact*** with L at p.

Next, we introduce the concept of *L-extension*. For any point p on L, let t be the longest vertical segment passing through p that intersects L only at points for which the segment is in right-contact with L. We easily see that there are three possible cases, depending or whether t has an endpoint on L, an unreached endpoint, or an endpoint at infinity. We define the *L-extension* of p as the segment t, with the convention that if t has an unreached endpoint, it is stretched to infinity past this endpoint. A point of L that lies on an L-extension is called a ***support-vertex*** if it is a vertex of L, and a ***pseudo-vertex*** otherwise. Note that an L-extension may be reduced to a single point or have both endpoints at infinity. In all cases, however, it contains a support-vertex.

The set of all L-extensions can be conveniently represented as a set of vertical trapezoids, with each vertical side coinciding with the L-extension of some vertex of L. This set defines the ***vertical decomposition*** of L, denoted $VD(L)$. If L is not closed, some of the trapezoids may be unbounded and there is no partitioning of space proper, since trapezoids may actually overlap. The set of trapezoids, however, induces a partition of the line L into line segments, called VD-edges. A VD-edge is a subsegment of an edge of L that ***sustains*** exactly one trapezoid. Note that a bounded (resp. unbounded) trapezoid is in general sustained by two (resp. one) VD-edges.

The data structure for $VD(L)$ provides two accessing schemes. On the one hand, each trapezoid is represented by its boundary, kept as a doubly-linked list, with adjacent trapezoids pointing to each other. On the other hand, the line L is represented as a doubly-linked list of VD-edges, each of them pointing to its sustained trapezoid. It may sometimes happen that the vertical side of a trapezoid contains several support-vertices. We resolve these singularities by introducing extra trapezoids of null width. This will allow us to assume that each trapezoid side contains exactly one support-vertex.

2.2 *Merging two VD-structures*

The key step of the divide-and-conquer algorithm involves merging two structures $VD(L_1)$ and $VD(L_2)$, with $L_1 = \{v_m, \ldots, v_1\}$, $L_2 = \{w_1, \ldots, w_m\}$, and $v_1 = w_1$. These lists indicate the orientation of the polygonal lines. For consistency, we use a descending sequence for L_1 so that the merge can proceed toward higher indices for both L_1 and L_2. In the following, we will say that a point $a \in L_1$ is *L_1-further* than a point $b \in L_1$ if it is encountered after b in the course of traversing L_1 from v_1 to v_m. The same applies about L_2. Let $L = \{v_m, \ldots, v_1, w_2, \ldots, w_m\}$. The goal is to compute $VD(L)$ by proceeding concurrently from v_1 to v_m and w_1 to w_m, reconfiguring the trapezoids on-line. The main feature of this *stitching* operation is to take *shortcuts* whenever possible, i.e. avoid traversing vertices whose adjacent trapezoids remain unchanged. This concept of shortcuts, extensively developed later on, is essential to the algorithm. Without it, all hope of beating $O(n \log n)$ time would vanish. The computation will be completely guided by two pointers, p_1 and p_2, running concurrently through the VD-edges of L_1 and L_2, respectively. The computation will ensure the following invariant:

Invariant: At any time during the merge of L_1 and L_2, the current state of $VD(L)$ consists of the L-extension of all the points of L_1 (resp. L_2) between v_1 and p_1 (resp. p_2) as well as the L_1- (resp. L_2-) extension of all the other points in L_1 (resp. L_2).

Figure 2 gives an example of a typical tour of p_1. For simplicity, we have represented L_2 as a curve line to give a rough indication of its interaction with L_1. The trajectory of p_1 appears as a dotted line. Figure 2 shows that p_1 will point to VD-edges of trapezoids in $VD(L_1)$, moving only between adjacent trapezoids. Occasionally, p_1 will switch from one VD-edge of a trapezoid to another VD-edge of the same trapezoid (recall that a trapezoid has one or two VD-edges). As we will see shortly, VD-structures contain all the information necessary to implement the merging without too much effort. The only difficulty resides in identifying all possible situations and treating them in a fairly unifying manner.

At the generic step, all the VD-edges of L_1 (resp. L_2) between v_1 and p_1 (resp. p_2) have been processed and will *never* be visited again during the merging of L_1 and L_2. If at time θ the last trapezoid t examined in $VD(L_2)$ has been entirely processed, we must consider the "next one" in $VD(L_2)$. This new trapezoid is referred to as the *fresh* trapezoid at time θ. More need be said about this trapezoid to make it defined unambiguously; all we can now say is that since it is adjacent to t, it can be retrieved in constant time from $VD(L_2)$. In general if the fresh trapezoid is provided by L_2, the last trapezoid, t', examined in $VD(L_1)$ will have been only partly updated at time θ, and further processing will be in order; t' is called the *carry* trapezoid at time θ. Typically, t' is the remaining part of a clipped trapezoid of $VD(L_1)$. By construction, the carry and fresh trapezoids lie on the same side —say, wlog, the left side— of the last trapezoid constructed. As far as the construction of the next trapezoid is concerned, the difference between a carry and a fresh trapezoid is immaterial. For this reason, we can assume wlog that p_1 stretches further to the left than p_2, i.e. the x-coordinate of p_1's left endpoint does not exceed the x-coordinate of p_2's left endpoint.

The algorithm rests on a case-analysis that can be best described pictorially. Figure 3 illustrates all possible cases. Each of the six pairs of pictures depicts the construction of the next trapezoid(s), represented in hashed lines in the right-hand column. Each of the twelve pictures contains an arrow to indicate the direction where the next construction will take place. We represent three kinds of trapezoids: *fresh* (F) and *carry* (C) as well as *next* (N), one of the trapezoids of $VD(L_2)$ adjacent to *fresh*. We will define *next* more precisely in each of the cases. The trapezoids are represented very schematically. Each of them has a VD- (lower or upper) edge currently under consideration; this edge is represented by a continuous line, whereas the other (lower or upper) edge is represented by a dotted line. The latter edge is either sustaining or is at infinity; in either case, its position has no particular significance, unless the line is doubled up with a continuous line (B,E). In this case, the edge is sustaining and its relative position is meaningful. The case-analysis is based on the relative position of p_2 ($= ab$) and bc, the VD-edge adjacent to p_2 in the direction of the construction. Cases (A,B,C) (resp. (D,E,F)) assume that c (resp. b) has the least x-coordinate among a, b, c. Here are a few scattered remarks intended to supplement Figure 3.

1. **(Fig.3-A,B)** – Let q be the unique support-vertex of F. From the Jordan Curve Theorem, it follows that either $q = b$, or q is strictly L_2-further than b. Figures 3-A,B assume the latter case. In the direction of the construction, there are two candidates for the role of next trapezoid from L_2. Discriminating between (A) and (B) settles this question. In (A), (resp. (B)), the support-vertex lies below (resp. above) p_1.
2. **(Fig.3-C,D,E,F)** – In all these cases, $q = b$, i.e. b is the support-vertex of the left side of F. Case (C) is the simplest and, as such, fairly self-explanatory. In (D), the left sides of both F and N extend to infinity downward, therefore no further interaction between L_1 and L_2 is to be expected: the merging is over. If this is not the case, let e_1 (resp. e_2) be the lower edge of C (resp. N). Note that one of these edges may be at infinity. Draw a segment from b downwards, until it first intersects either e_1 (case E) or e_2 (case F). In

the former case, let N' be the trapezoid of $VD(L_2)$ sustained by bc. We can process directly all of C and N, declare N' fresh and keep C as the carry. Two observations are in order: first of all, although C is still a carry trapezoid, its sustaining edge that gives its value to p_1 will be switched from the upper to the lower edge. Also, it may be the case that N is not properly defined as indicated in Fig.3-E. This will happen when the trapezoid of $VD(L_2)$ sustained by bc is unbounded below. We then blithely proceed towards N', after applying the treatment depicted in Fig.3-E. Finally, in case (F), we proceed as shown in the corresponding figure, observing that the treatment is somewhat fairly similar to (A,C).

We omit the proof of correctness.

2.3 The Divide-and-Conquer Procedure

We are now ready to compute the vertical decomposition of the polygon P. Let $\{p_1, \ldots, p_n\}$ be the vertices of P in clockwise order. Recursively, compute $VD(L_1)$ and $VD(L_2)$, for $L_1 = \{p_1, \ldots, p_{\lfloor n/2 \rfloor}\}$ and $L_2 = \{p_{\lfloor n/2 \rfloor + 1}, \ldots, p_n\}$, and finally apply the procedure described above to merge $VD(L_1)$ and $VD(L_2)$. This will produce the vertical decomposition of P, which completes the description of the algorithm. Here is a simple, preliminary result on the complexity of the algorithm:

Theorem 1. The algorithm computes the vertical decomposition of a simple n-sided polygon in $O(n \log n)$ time and $O(n)$ space.

Proof: Omitted. ▮

We will see later on that the algorithm has a much better performance than indicated by Theorem 1. To seek any improvement, however, it is crucial to ensure two requirements regarding the implementation.

1. If the algorithm is implemented recursively, we should pass pointers to the endpoints of L_1 and L_2 as arguments to the merge routine VD and not the full-fledged lists themselves.

2. We should compute $VD(L)$ *in place* by modifying the existing links; allocating new space only when necessary, and "garbage-collecting" after deletions.

3. The Triangulation Algorithm

We will show how to transform the vertical decomposition of P into a triangulation, using a variant of the algorithm of Garey et al [G]. This algorithm involves computing a *regular decomposition* of P and applying to it a simple $O(n)$ time procedure to produce a triangulation. A regular decomposition of a polygon P is a partition of P into *monotone* polygons that does not introduce new vertices. A polygon Q is said to be *monotone* if there exists a line T such that the boundary of Q can be decomposed into two consecutive chains, each of whose orthogonal projections on T has the same vertex-list order as its originating chain.

It is quite simple to go from a vertical to a regular decomposition of P. To do so, we choose the x-axis as the reference line T, and identify all the vertices of P that cause the polygon not be monotone. These are exactly the support-vertices whose P-extension contains two pseudo-vertices. For each such vertex v, let t be the unique trapezoid for which the P-extension of v coincides with a vertical side, and let w be the other support-vertex of t. It is easy to show that connecting each pair (v, w) in P by a straight-edge will produce a regular decomposition of P (Fig.4). We omit the proof of this elementary fact. The description of the triangulation algorithm is now complete. Its time complexity is clearly $O(n \log n)$, since transforming a vertical decomposition into a triangulation requires linear time.

4. Introducing the Notion of Nesting Number

Since the complexity of the triangulation algorithm is entirely dominated by the construction of the vertical decomposition, we will concentrate exclusively on this phase of the algorithm. A little observation shows that the upper bound of $O(n \log n)$ on the time complexity of the algorithm is tight (Fig.5-A). The shapes of the worst-case polygons are so contrived, however, that we can argue that in most practical cases, the algorithm is *extremely* efficient —actually more efficient than any other triangulation algorithm known to date. This feature is owed to the fact that the algorithm is highly adaptive. Indeed, by constantly seeking shortcuts, the algorithm may often merge two large inner decompositions in constant or near-constant time. In Figure 5-B, for example, all the merges take constant time, except for the last one which requires linear time. The overall running time is therefore $O(n)$. We will in this section try to characterize classes of polygons for which the running time is linear or quasi-linear.

To refine the crude analysis of Theorem 1, we introduce the concept of *nesting number*. A vertical line D that intersects P has the effect of dividing the boundary of P into polygonal chains running from one point of D to another, each lying entirely on one side of D. Consider now the polygonal lines as fences and imagine that a dog placed initially on the line D wishes to run away to infinity staying always on one side of D. Let m_1 (resp. m_2) be the minimum number of fences it will have to jump in order to escape through the left (resp. right) side, and let $m = \max(m_1, m_2)$. We define $\nu(P)$, the *nesting number* of P, as the maximum value of m for all starting positions and all lines D. In Figure 6, for example, $\nu(P) = 2$. Practical experience (with, say, graphical typesetting or pattern recognition, where the need for efficient triangulations arises) shows that the nesting number of a polygon is in general one and most often less than or equal to two or three. Let's assume in the following that $\nu(P) \leq 3$; we will establish an interesting result on the complexity of the triangulation algorithm for this class of polygons.

It is clear from the description of the algorithm that the running time is proportional to the number of times some L-extension is updated. Pseudo-vertices are never updated, properly speaking; they are created and possibly destroyed. Consequently, the costs which they incur can be charged to the corresponding support-vertices. Wlog, we can concentrate on the L-extensions that have exactly one pseudo-vertex lower than their support-vertex along the y-axis. These extensions are called *top-extensions*, and their lower endpoints (which are pseudo-vertices) are called the *low-points* of the extensions. Let $C(n)$ be the maximum number of times some low-point is updated in the course of computing $VD(P)$. By symmetry, $C(n)$ will also give an upper bound on the number of updates involving the other type of L-extensions.

Since a low-point can never be set to infinity twice in a row, we may reset $C(n)$ to count only the times when a low-point is set to a position on the boundary of P. These updates are called *breaks*. Let p_i be the support-vertex of some top-extension. Note that chronologically, breaks occur closer and closer to p_i. The vertex p_i is said to be *right-broken* (resp. *left-broken*) if the break is caused by an edge $p_j p_{j+1}$ with $j > i$ (resp. $j < i$).

Let's examine how left-breaks can occur in the course of merging the vertical decomposition of two polygonal lines L_1 and L_2. Wlog, assume that $L_1 = \{p_1, \ldots, p_m\}$ and $L_2 = \{p_{m+1}, \ldots, p_{2m}\}$. For any $k > 2$, let $D(k)$ be the number of vertices in L_2 that satisfy the three following conditions: 1) they are left-broken in the course of merging L_1 and L_2 (this can happen only once per merge), 2) they were previously left-broken exactly $k-1$ times, and 3) they will be left-broken later on at least once more. We can prove the following result (the proof is too technical to be included).

Lemma 1. For any $k > 2$, we have $D(k) \leq m/2^{k-3}$.

We conclude.

Theorem 2. The triangulation algorithm runs in $O(n \log\log n)$ time for any n-sided polygon with nesting number less than 4.

Proof: Omitted. ▮

5. Preconditioning the Input

As is often the case with adaptive algorithms, preprocessing the input prior to calling the main algorithm might often lead to substantial gains. One advantage of the algorithm is that it is based on polygonal lines rather than polygons. This features allows us to break down the polygon into a (preferably small) number of polygonal lines, the vertical decompositions of which we will compute separately and merge together in a final stage. We will take advantage of the fact that the algorithm is particularly efficient on spiral-shaped inputs.

Let $L = \{q_1, \ldots, q_m\}$ be a simple polygonal line. Consider the motion of the straight-line passing through $q_i q_{i+1}$, as i goes from 1 to $m-1$. Every time the line reaches the vertical position clockwise (resp. counterclockwise), we increment (resp. decrement) a *winding counter* by one (the initial value of the counter is irrelevant). We will say that L is *spiraling* (resp. *anti-spiraling*) if the winding counter is never decremented (resp. incremented) twice in a row.

It is straightforward to decompose any simple polygon into spiraling and anti-spiraling polygonal lines. This can be done in a single linear pass, and we may omit the details. Note that we should restart a new polygonal line only when the previous line ceases to be spiraling or anti-spiraling (Fig.7). At that point, we start scanning the new polygonal line, and continue the traversal as long as the line can be classified as either spiraling or anti-spiraling. In general, the line will be of both types at the start and then fall into one type. We define s, the ***sinuosity*** of P, as the number of polygonal lines thus obtained. Note that depending on the starting vertex, this number may vary by one. In the following, s will refer to the larger of these numbers. Empirical evidence shows that in practice s is a small constant. Even a fairly contrived polygon like the one depicted in Figure 7 has a sinuosity of only 5. We will show successively that the vertical decomposition of a spiraling or anti-spiraling line can be computed in linear time, and that a simple n-sided polygon P can be triangulated in time $O(n \log s)$.

Lemma 2. The vertical decomposition of any spiraling or anti-spiraling polygonal line can be computed in linear time.

Proof: Omitted. ▮

We can now state the main result of this section.

Theorem 3. It is possible to triangulate a simple n-sided polygon P in $O(n \log s)$ time and $O(n)$ space.

Proof: Based on the construction of an optimal alphabetic tree, using the Hu-Tucker algorithm [K]. ▮

6. Conclusions

We have presented a new triangulation algorithm for simple polygons that contains a number of attractive features, both of theoretical and practical nature. On the practical side, our algorithm has been implemented and has shown to be, as expected, extremely efficient. In graphics, a common application of polygon triangulation is region-filling, whereby a given area is to be *painted*. It turns out that with modern raster graphics devices, painting a vertical trapezoid can be accomplished just as fast as painting a triangle, so we may use the vertical decomposition algorithm directly without resorting to the triangulation, which is bound to save substantial CPU time. On the theoretical side, we have established that the algorithm runs in $O(n \log s)$ time, where n is the input size and s is the ***sinuosity*** of the polygon. The latter quantity is in general extremely small, but even in most of the cases where it is large, our algorithm, being highly adaptive, will behave linearly or quasi-linearly. We have also exhibited a large class of polygons for which the algorithm runs in time $O(n \log \log n)$. Identifying exactly the class of polygons for which the algorithm runs in linear time is a challenging, yet very worthwhile endeavor; indeed, we conjecture that this class is considerably larger than the subset exhibited in this paper. The theoretical implications of our results are, in particular, the evidence that in general triangulating is easier than sorting, although, in the worst case, no such statement can yet be made. Deciding on this issue once and for all is undoubtedly one of the most outstanding open questions of computational geometry.

REFERENCES

[C] Chazelle, B. *A theorem on polygon-cutting with applications*, Proc. 23rd IEEE Symposium on Foundations of Computer Science, Chicago, Nov. 1982.

[G] Garey, M.R., Johnson, D.S., Preparata, F.P. and Tarjan, R.E. *Triangulating a simple polygon*, Info. Proc. Lett., Vol. 7(4), June 1978.

[H] Hertel, S., Mehlhorn, K. *Fast triangulation of simple polygons*, Proc. FCT'83, 1983.

[K] Knuth, D.E. *Fundamental algorithms*, Vol.1, Addison-Wesley.

[S] Shamos, M.I. *Computational geometry*, PhD thesis, Yale University, 1978.

[SV] Schoone, A.A., van Leeuwen, J. *Triangulating a star-shaped polygon*, Tech. Report, RUV–CS–80–3, Univ. of Utrecht, April 1980.

[T] Toussaint, G.T. *Pattern recognition and geometrical complexity*, 5th Int. Conf. on Pattern Recognition, Dec. 1980.

[TA] Toussaint, G.T., Avis D. *On a convex hull algorithm for polygons and its application to triangulation problems*, Pattern Recognition, Vol. 15(1), pp. 23–29, 1982.

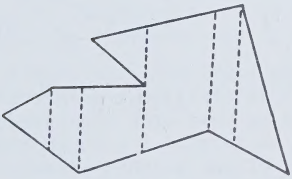


Figure 1

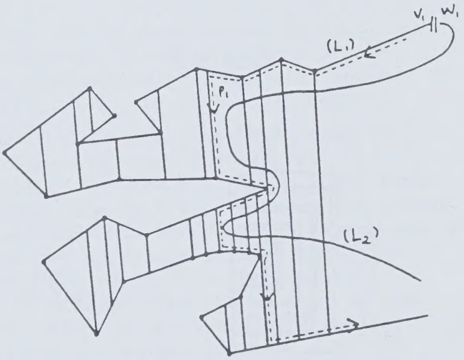


Figure 2

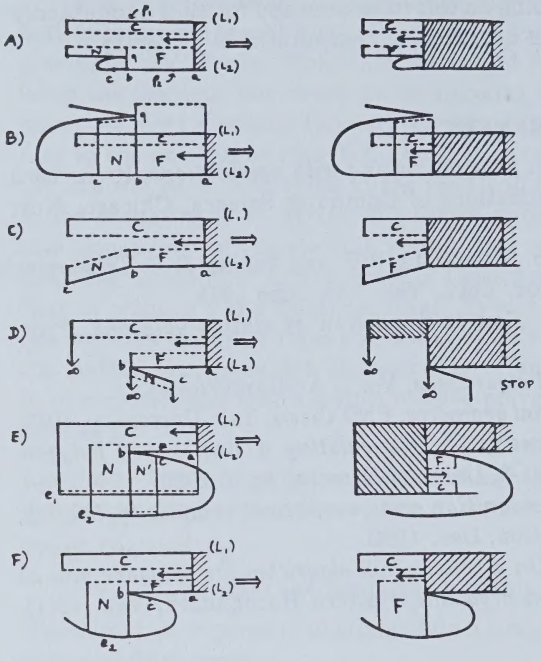

A)
B)
C)
D)
E)
F)
P_1
C
N
F
q
(L₁)
(L₂)
c
b
P_2
a
∞
STOP
N'
e_1
e_2

Figure 3

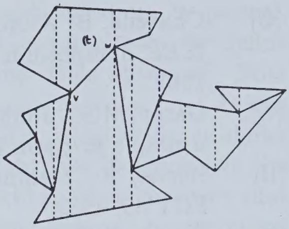


Figure 4

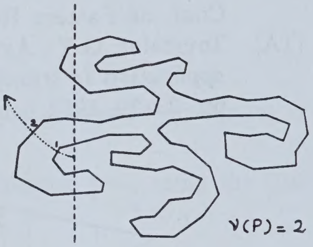


Figure 6

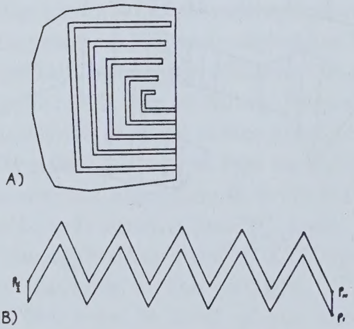


Figure 5

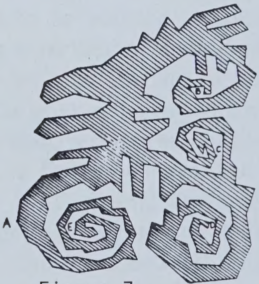


Figure 7

AN OPTIMAL ALGORITHM FOR COMPUTING THE MINIMUM VERTEX DISTANCE BETWEEN TWO CROSSING CONVEX POLYGONS

GODFRIED T. TOUSSAINT
McGill University, Montreal, Quebec, Canada

Let $P = \{p_1, p_2, \ldots, p_m\}$ and $Q = \{q_1, q_2, \ldots, q_n\}$ be two intersecting convex polygons whose vertices are specified by their cartesian coordinates in order. An optimal $O(m + n)$ algorithm is presented for computing the minimum euclidean distance between a vertex p_i in P and a vertex q_j in Q.

INTRODUCTION

Let $P = \{p_1, p_2, \ldots, p_m\}$ and $Q = \{q_1, q_2, \ldots, q_n\}$ be two convex polygons whose vertices are specified by their cartesian coordinates in clockwise order. We assume the polygons are in *standard* form, i.e., no three consecutive vertices are collinear. Let $d(x,y)$ denote the euclidean distance between points x and y. Considerable attention has been given recently to the problem of computing extremal distances between convex polygons due to their application in pattern recognition and collision avoidance problems [1], [2]. One such problem consists of finding the *minimum* distance between the polygons, i.e., zero if the polygons intersect and the minimum distance $d(x,y)$ realized by a pair of points $x \in P$, $y \in Q$, if P and Q do not intersect. Edelsbrunner [1] describes an optimal $O(\log m + \log n)$ algorithm for solving this problem. This improves an earlier algorithm for this problem due to Schwartz [2] which runs in $O((\log m)(\log n))$ time.

A more difficult problem is to find the *minimum vertex distance* between P and Q, i.e., the minimum distance $d(x,y)$ where x and y are restricted to being vertices of P and Q, respectively. The naive method of computing distances $d(p_i, q_j)$ for all i and j requires, of course, $O(mn)$ time. By computing supergraphs of the minimal spanning tree of the union of the vertices of P and Q Toussaint and Bhattacharya [3] have shown that this problem can be solved in $O((m + n) \log(m + n))$ time. The methods of [3] however do not exploit the fact that P and Q are convex.

Recently McKenna and Toussaint [4] and Chin and Wang [5] independently discovered optimal $O(m + n)$ algorithms for solving this problem in the special case where P and Q are *linearly separable*, i.e., the polygons do not intersect. The algorithm in [4] differs from that in [5] in that it is based on existing results on the relative neighborhood graph [6]. With trivial modifications the algorithms in [4] and [5] also work if only the edges of P and Q intersect, i.e., as long as the interiors of the polygons do not intersect.

In this paper we show that when the interiors of P and Q intersect the minimum vertex distance can also be computed in $O(m + n)$ time. The problem is split into two cases: the case when one polygon is completely contained in the other and the case where this is not true. The key result for obtaining a solution to both cases consists of decomposing a convex polygon into parts associated with regions on the plane where the Voronoi diagram can be computed in linear time.

MAIN RESULTS

Theorem 1:

Given a convex polygon P_s of n sides, with the *semi-circle* property [7] with respect to edge $p_i p_{i+1}$, then the Voronoi vertices of the Voronoi diagram of P_s all lie in P_s or the right-half plane determined by p_ip_{i+1}.

Theorem 2:

The minimum vertex-distance between two convex polygons P and Q of m and n vertices, respectively, can be computed in $O(m + n)$ time.

REFERENCES

[1] Edelsbrunner, H., "On computing the extreme distances between two convex polygons", technical report F96, Technical University of Gratz, 1982.

[2] Schwartz, J. T., "Finding the minimum distance between two convex polygons", *Information Processing Letters*, 1981, pp. 168-170.

[3] Toussaint, G. T., and B. K. Bhattacharya, "Optimal algorithms for computing the minimum distance between two finite planar sets", *Fifth International Congress of Cybernetics and Systems*, Mexico City, August 1981.

[4] M. McKenna and G. T. Toussaint, "Finding the minimum vertex distance between two disjoint convex polygons in linear time", Tech. Report No. SOCS - 83.6, School of Computer Science, McGill University, April 1983.

[5] Chin, F., and Wang, C. A., "Minimum vertex distance problem between two convex polygons", technical report, University of Alberta, 1983.

[6] Supowit, K. J., "The relative neighborhood graph, with an application to minimum spanning trees", *J.A.C.M.*, in press.

[7] D. T. Lee and F. P. Preparata, "The all-nearest-neighbor problem for convex polygons", *Information Processing Letters*, Vol. 7, 1978, pp. 189-192.

ON THE DUALITY OF INTERSECTIONS AND CLOSEST POINTS (SUMMARY)

Chanderjit Bajaj and Ming Li

Department of Computer Science,
Cornell University,
Ithaca, NY 14853.

ABSTRACT: We call the common intersection of k objects a k-intersection. We address questions of the form: Given n objects in the plane, 'What is the maximum value of k for which there exists a k-intersection ?'. In answering such questions we utilize a *duality* that exists between k-intersections and k-closest points. A technique, dubbed 'rotation sorting' is developed to provide efficient solutions to dual questions of the form: Given n points in the plane, 'What is the maximum number of points that can be enclosed by a circle of radius r ?'. A sorting of the points of a set is defined with respect to a single fixed point x of this set, with other points being ordered in the way they encouter the arc of a circle of radius r as the arc is rotated about x. Another intersection planar graph technique [Hop83] is also used to deal with the unequal radius circles. A similar duality between the intersection of a line with circles and the proximity of this line to the centers of the circles is exploited to obtain efficient algorithms for the transversals of circles in the plane. Extensions of the above questions to three and higher dimensions are also addressed alongwith problems concerning unequal sized objects. Further generalisations of k-intersections and of the number of points enclosed by k objects are shown to be NP-complete and D^p-complete. For the proofs and the references, the readers are referred to [1].

1. Developing the Duality

Let c_i be the center of the i^{th} circle and let d(x,y) = Euclidean distance between any two points x and y. Let the minimum enclosing circle of the centers $\{c_i\}$ of k circles in the plane be denoted by MEC^k, i.e. if c = center of $MEC^k \Rightarrow \forall$ i=1..k, d$(c,c_i) \leq$ radius of MEC^k.

Theorem 1: For k circles of equal radius r, the k-intersection is non-null **iff** $r \geq$ radius of MEC^k. This gives rise to a *duality* which exists between k-intersections and k-closest points.

Corollary: For k circles of unequal radius r_i, i=1..k, the k-intersection is non-null **if** Minimum $\{r_i\} \geq$ radius of MEC^k.

2. Intersections and Closest Point Problems

Given n circles of equal radius r in the plane,

(p1) [*Detection of a* k-intersection] Does there exist a k- intersection ?

(p2) [*Counting* k-intersections] How many k-intersections exist ?

(p3) [*Maximum* k-intersection] Find the maximum value of k for (p1).

(p4) [k-*closest points*] Find k points amongst n points in the plane, with the smallest enclosing circle.

---

[1]This research was supported in part by NSF grants MCS81-01220, 8301766.

Theorem 2: (1-Dim case) Given n intervals of equal length r on a line to find the maximum number k of intervals having a common intersection takes Ω (nlogn) time. And the problem can be solved in O(nlogn) time.

Theorem 3: Problems p1 - p3 can be solved in $O(n^2 logn)$ time, or by n processors in O(nlogn) parallel time. This is also true for unequal sized circles. (A more precise time bound is $O(\sum_{i=1}^{n}(n+ m_i logm_i))$ where m_i=# of intersections on circle i.)

Corollary : For n spheres in m-dimensions The maximum k-intersection problem can be solved in $O(n^m \text{logn})$ time.

For the k-closest points problem we use what is known as the order k-nearest neighbor Voronoi diagram. [Shamos75],[Lee82]. From [Lee82] we know that $V_k(S)$ can be obtained in $O(k^2 nlogn)$ time using $O(k^2(n-k))$ storage.

Theorem 4: The k-closest points problem can be solved in $O(k^2 nlogn)$) time.

Corollary : The k-intersection detection problem can be solved in O(minimum{$O(k^2 nlogn)$,$O(n^2 logn)$} time.

3. Transversal problems

(p5) [*Circular Transversal*] Does one of the given circles intersect all the others ?

(p6) [*Line Transversal*] Does there exist a line which intersects all the circles ?

Theorem 5: The circular transversal problem for n circles of equal radius r in the plane can be solved in $O(nlogn)$ time.

Corollary : Given n circles of equal radius r, one can determine whether any circle of radius R with center c is a circular transversal in $O(logn)$ time using $O(nlogn)$ preprocessing.

Corollary : Given n points in the plane, one can determine whether any circle of radius R with center c encloses all the points in $O(logn)$ time using $O(nlogn)$ preprocessing.

Theorem 6: The line traversal problem for n circles of equal radius r in the plane can be solved in $O(nlogn)$ time.

4. NP-complete and D^p-complete problems

Theorem 7: **[Piercing number]** Given n circles of radius r in the plane, the problem of deciding whether k needles can pierce all n circles is NP-complete.

Theorem 8: **[k-circle Max-m]** Given n points on plane and k (relocatable circles of radius r, the problem of deciding whether the maximum number of points covered by k circles $= m$ is D^p-complete.

Corollary : The dual problem of the maximum piercing number is also D^p-complete.

Theorem 9: *[All points Min-k* **circle]** Given n points in the plane and k (relocatable) circles of radius r that the minimum number of circles that cover all points $= k$ is D^p-complete.

5. Reference

[1] C. Bajaj, and M. Li, *On the Duality of intersections and closest points.* TR 83-568 Dept. of Computer Science, Cornell University, 1983

AN $O(|V|)$ PARALLEL ALGORITHM TO CONSTRUCT MINIMAL SPANNING TREES BASED UPON VORONOI DIAGRAMS

R. C. CHANG
Institute of Computer Engineering, National Chiao Tung University
Hsinchu, Taiwan 300, Republic of China

R. C. T. LEE
Institute of Computer and Decision Sciences
National Tsing Hua University
Hsinchu, Taiwan 300, Republic of China

ABSTRACT — In this paper, we shall present a parallel algorithm to construct minimal spanning trees out of a set of 2-dimensional points. This algorithm is based upon the concept of Voronoi diagrams. If implemented in parallel, its time complexity is O(N) and it requires O(logN) processors where N is the number of input points.

SECTION 1. INTRODUCTION

In this paper, we are concerned with algorithms to construct minimal spanning trees. The minimal spanning tree problem can be roughly stated as follows. We are given a set of points. Our job is to connect them into a tree such that its total length is minimized.

There are many problems which can be solved by the minimal spanning tree concept. Prim [1957] showed that building an optimal telephones network needs finding a minimal spanning tree. Loberman and Weinberger [1957] applied the minimal spanning tree concept to connect terminals on a breadboard. Minimal spanning trees are also useful in pattern recognition [Zahn 1971] clustering analysis [Gower and Ross 1969, Johnson 1967, Lee 1981]. Lee, Slagle and Blum [1977] showed that minimal spanning trees can be used to perform a high-dimensional mapping. Kang, Lee, Chang and Chang [1977] applied minimal spanning trees to reduce storage requirement.

There are many algorithms to construct minimal spanning trees. [Bentley 1980, Bentley and Friedman 1978, Cheriton and Tarjan 1976, Chin, Lam and Chen 1982, Deo and Yoo 1981, Dijkstra 1958, Kruskal 1956, Nath and Maheshwar 1982, Prim 1957, Savage and Ja' Ja' 1981, Shamos 1978, Sollin 1977, Tarjan 1982, Yao 1975, Yao 1982]. For most algorithms to construct minimal spanning trees, the input data is a graph. We shall assume that our data are a set of two-dimensional points. Bentley and Friedman [1978] and Shamos [1978] took the same point of view.

Our research arises from one puzzling question: Can a minimal tree on a plane can be formed by divide-and-conquer approach? Note that a minimal spanning tree connects a set of points into a tree. Since the total length of this tree is to be minimized, we expect two points to be connected if they are relatively close to each other. In other words, if there are several points between two points P_1 and P_2, then it is quite unlikely that there will be an edge connecting P_1 and P_2 in a minimal spanning tree. This implies that the calculation of the distance between P_1 and P_2 should be avoided. Bentley and Friedman [1978] discussed this point. They pointed out that in implementing a minimal spanning tree algorithm, nearest neighbor searching [Bentley, Weide and Yao 1980, Burkhard and Keller 1973, Friedman, Baskett and Shustek 1975, Friedman, Bentley and Finkel 1977, Fukunaga and Narenda 1975, Lee 1981, Lee, Chin and Chang 1976, Shen and Lee 1978], we can avoid many distance calculations. For the case mentioned above where P_1 is relatively far away from P_2, we can probably avoid the distance calculation between them if a good nearest neighbor searching concept is incoorperated into the minimal spanning tree construction algorithm, as it was in [Bentley and Friedman 1978]. That a minimal spanning tree can be formed by using the nearest neighbor searching approach indicates that local information is crucial and divide-and-conquer approach may therefore be appropriate. In [Shamos 1978], it was pointed out that a minimal spanning tree can be extracted from a Voronoi diagram [Shamos 1978, Voronoi 1908]. In this paper, we shall go one step further. We shall show that a minimal spanning tree can be constructed as a Voronoi diagram is being constructed. Since a Voronoi diagram can be constructed by a divide-and-conquer approach, a minimal spanning tree in a two-dimensional plane can be constructed by this same approach.

Because our algorithm uses the divide-and-conquer approach, our algorithm can be implemented as a parallel algorithm. Denote $|V|$, the number of nodes, to be N. The complexity of our parallel algorithm is O(N), using O(logN) processors.

Limited by space, we can not describ the concept of Voronoi diagram here. We have to assume that the reader is familiar with Voronoi diagrams [Voronoi 1908, Leob 1976, Marden 1966, Hodder and Orton 1976, Shamos and Hoey 1975, Shamos 1978].

In the next section, we shall show that minimal spanning tree can be constructed in a divide-and-conquer manner.

SECTION 2. THE DIVIDE-AND-CONQUER APPROACH TO CONSTRUCT MINIMAL SPANNING TREES

That a minimal spanning tree for a set of points in the Euclidean plane can be constructed by a divide-and-conquer approach is based upon the following observation. Suppose we use a straight line L to divide these points into V_L and V_R. We can construct the minimal spanning trees for V_L and V_R respectively. They are denoted as MST(V_L) and MST(V_R). If we try to merge these two minimal spanning trees, what will happen? Obviously, the leftmost part of MST(V_L) and the rightmost of MST(V_R) will hardly be affected by this merging. They will probably remain unchanged. The only parts which will be affected by this merging process are the central parts of MST(V_L) and MST(V_R). Thus, if there is a correct way to merge MST(V_L) and MST(V_R) efficiently, then a minimal spanning tree can be constructed by using a divide-and-conquer approach.

It turns out that there is indeed a merging process which is efficient. Note that a minimal spanning tree is a part of the Delaunay triangulation, as shown in [Shamos 1978]. To merge MST(V_L) and MST(V_R), we merely examine the Delaunay triangulation edges between MST(V_L) and MST(V_R) to see whether they should be added to connect these two minimal spanning trees. We shall show later that the algorithm to find Delaunay triangulations proposed in [Lee and Schachter 1980] can be used as the merging process because it will find all of the Delaunay edges between MST(V_L) and MST(V_R).

Since a minimal spanning tree is a connected graph, points in V_L must be connected to points in V_R. In other words, edges must be added. If more than one edge is added, then a cycle is formed and some edge in this cycle must be deleted to keep the resulting graph a tree.

Thus our merging process involves two kinds of actions: adding new edges and possibly deleting edges. Because all minimal spanning tree edges must be Delaunay edges [Shamos 1978], we only have to consider edges which are Delaunay edges. To determine whether an edge is a Delaunay edge or not, we may use the method proposed in [Lee and Schachter 1980].

The algorithm proposed by Lee and Schachter [1980] was originally designed for merging two Delaunay triangulations. Here, we may simply view it as an algorithm to find a candidate set of Delaunay edges.

As pointed out in [Lee and Schachter 1980], there is an ordering of candidate edges.

The merging process adds the candidate edges one by one from bottom to top. If a cycle is formed after adding an edge, the longest edge in cycle is deleted. For the minimal spanning trees in Fig. 2-1, (v_3,v_{10}) is the first edge to be added. After (v_6,v_8) is added, a cycle is formed. In this cycle, (v_3,v_{10}) is the longest and is thus deleted. Finally, (v_5,v_7) is added.

Fig. 2-1 here

A cycle is again formed. We find out that (v_5,v_8) is the longest in this cycle. So (v_5,v_8) is deleted. The resulting minimal spanning tree is shown in Fig. 2-2.

Fig. 2-2 here

Having described the algorithm informally, we are now ready to present it formally. Our algorithm is called MINISPAN which is a recursive algorithm whose input is a set V of N points in the plane and whose output is a minimal spanning tree for V.

Algorithm MINISPAN

Input : A set V of N points in the Euclidean plane.
Output: A minimal spanning tree of V.
Step 1: Choose a cut line L perpendicular to the X-axis such that N/2 points of V have X-values less than L and N/2 points have X-values greater than L. These two point sets are denoted as V_L and V_R respectively.
Step 2: Recursively find the minimal spanning trees of V_L and V_R.
Step 3: Use the algorithm proposed by Lee and Schachter [1980] to find a candidate edge set, which is a subset of the Delaunay edges of V. Add the edges in the candidate edge set into MST(V_L) and MST(V_R) one by one from bottom to top. If a cycle is formed, delete the longest edge in the cycle.

Some observation about our algorithm is now in order. First of all, our algorithm is easy to be comprehended and hand simulated. At the very beginning of the algorithm, the algorithm merely connects two points into an edge. Later, Delaunay edges are found and added one by one. Since Delaunay edges are quite natural in concept, they can be found visually. Whenever a cycle is found, the longest link, which is easy to find, it deleted.

Another important property of our algorithm is that unnecessary distance calculations are avoided.

SECTION 3. THE CORRECTNESS OF MINISPAN

In this section, we shall prove the correctness of MINISPAN. We shall use the following result proved in [Tarjan 1982].

Lemma 3.1 [Tarjan 1982]. T is a minimal spanning tree if and only if for each non-tree edge of (v,u), the length of (v,u) is at least as long as the length of any edge on the unique cycle in T formed by joining v and u.

Using the above lemma, we can prove the following:

Theorem 3.1 Algorithm MINISPAN produces a minimal spanning tree of V as its output.

Proof: Since we always delete one edge when a cycle is formed during the merging process, the resulting graph T is a spanning tree of V. In the sequel, we shall show that the spanning tree is also minimal.

Suppose that T is not a minimal spanning tree. Then there exists an edge $(v,u) \in T$ such that the length of (v,u) is shorter than some edge on the cycle by joining v and u.

The two vertices of edge (v,u) must belong to one of the following two classes:

Class 1: $u, v \in V_L$ or V_R.

Class 2: $v \in V_L$, $u \in V_R$ (or $v \in V_R$, $u \in V_L$).

If (v,u) belongs to Class 1, then either $(v,u) \in$ {edges deleted during the merging steps} or $(v,u) \in \{E_L - MST(V_L)\}$ where E_L is the edge set of V_L. But the edges deleted during the merging steps are the longest ones on each cycle formed in the merging steps of MINISPAN. Therefore $(v,u) \notin$ {deleted edges}.

Assume that $(v,u) \in \{E_L - MST(V_L)\}$, then adding (v,u) generates two kinds of cycles, one consisting of edges only in $MST(V_L)$ and the other including some edges in V_R. For the former case, since $MST(V_L)$ is a minimal spanning tree of V_L, according to Lemma 3.1, (v,u) must be the longest edge in the cycle. The situation of the latter case can be described in Fig. 3.1. In this case, $e_1, e_2, \ldots, e_k \in E_L$ and $e'_1, e'_2, \ldots, e'_j \in E_R$.

Fig. 3-1 here

Since $MST(V_L)$ is connected, there must exist a deleted edge (u',v') which is an edge of a cycle consisting of only edges of E_L. Since $(u',v') \in MST(V_L)$, according to Lemma 3.1, the length of (u,v) is greater than or equal to that of (u',v'). But (u',v') is deleted only if (u',v') is the longest in the cycle formed during the merging steps. Therefore, (v,u) is the longest edge in the cycle consisting of $(u,v), e_1, e_2, \ldots, e'_1, e'_2, \ldots, e'_j, \ldots, e_k$.

Suppose $(v,u) \in$ Class 2. For algorithm MINISPAN, an edge in Class 2 is in T as long as it is not the longest edge in any cycle formed during the merging steps. Therefore $(v,u) \notin$ Class 2. In conclusion, (v,u) can not exist and according to Lemma 3.1, T must be a minimal spanning tree of V.

Q.E.D.

SECTION 4. THE COMPLEXITY ANALYSIS OF MINISPAN

Let us first assume that the computation model is a random access machine (RAM) as described in Aho, Hopcroft and Ullman [1974].

Step 1 can be executed by using the linear median algorithm proposed by Blum, Floyd, Pratt, Rivest and Tarjan [1972] in time $O(N)$ in the worst case. Since Step 2 solves two problems of size $N/2$, the cost of this step is $2 \cdot O(N/2)$. The timing of Step 3 needs a deeper analysis. We claim that Step 3 takes $O(N)$ time in the worst case, as indicated in the following Theorem.

Theorem 4.1

The merging step in Algorithm MINISPAN takes $O(N)$ in the worst case.

Proof: Finding the candidate set of edges takes $O(N)$ operations (Lee and Schachter 1980]. Since a Delaunay triangulation contains at most $O(N)$ edges, the maximum number of candidate edges is $O(N)$. The remaining problem is whether the total number of edges traversed when cycles are formed is also $O(N)$. Since the total number of edges in a cycle formed in the merging step is $O(N)$, a brute force implementation of Step 3 will take $O(N^2)$ steps in the worst case. In the following, we shall show that edges in $MST(V_L)$ and $MST(V_R)$ do not have to be traversed more than twice. Note that we add edges to the MST one by one and from bottom to top. If an edge (u,v) is added and a cycle C is formed as shown in Fig. 4-1, there are two possibilities:

Case 1: (u,v) is the longest in C.

Case 2: (u,v) is not the longest in C.

In case 1, (u,v) is deleted. The next cycle will include all e_i's. However, we do not have to traverse them again because we only have to record the longest edge among e_i's and compare the new candidate edge with this edge without cycle traversing. That means, we only have to traverse the edges once in this case.

Fig. 4-1 here

Case 2 is more complex. In this case, (u,v) is retained and the longest edge in current cycle is deleted. If another cycle is formed, then the new cycle will include some edges between e_1 and e_L, (u,v), but not any edge e_i, $L+1 \leq i \leq k$. Thus for e_i, $L+1 \leq i \leq K$, the edges are traversed only once in this case. For (u,v) and e_i, $1 \leq i \leq L$, these edges will be traversed again in the next cycle. However, these edges will be traversed only once in the next cycle. Thus each edge will be traversed at most twice.

Therefore, We conclude that the merging steps take O(N) in the worst case.

Q.E.D.

Assume that the total running time of MINISPAN is T(N). Based on Theorem 4.1, we can formulate the following recurrence relation:

$$T(N) = 2T(N/2) + O(N)$$

The solution of T(N) is O(NlogN). Therefore, the running time of MINISPAN is O(NlogN).

Algorithm MINISPAN can readily been implemented on a tree machine, [Bentley and Kung 1979, Ottman, Rosenberg and Stockmeyer 1982], as depicted in Fig. 4-2.

Fig. 4-2 here

To simplify our discussion, let us denote the bottom of the tree machine to be the first level, which contains logN processors, the ancestors of the first level as the second level. In the same fashion, the ancestors of the kth level is the (k+1)th level. Accordingly, the root of the tree machine is in the loglogN level, and each processor can be numbered as PU_{ij}, where i is the level number, and j is the sequence number of the same level (from left to right). For example PU_{14} denotes the fourth processing unit in the first level. Assuming that we have a drive computer, which feeds data into bottom level of processors, a minimal spanning tree can be constructed in parallel.

Firstly, we feed N data points into the processors in the first level. This takes N operations. Then, each processor PU_{1j}, j=1,...,logN, finds in parallel the (N/logN) data points which fall between $(N/\log N)\cdot(i-1)$th and $(N/\log N)\cdot i-1$th (according to X-axis) points respectively. This also can be accomplished in O(N) by using the linear median algorithm [Blum, Floyd, Pratt, Rivest and Tarjan 1972]. The third step is to find the minimal spanning trees of these (N/logN) points in the first level in parallel. This takes $O((N/\log N)\log(N/\log N)) \leq O(N)$ by MINISPAN. The output of each PU_{1j} is a minimal spanning tree of the (N/logN) points. Applying the merging step of MINISPAN, we can merge these trees in the subsequent levels of processors in parallel. In the worst case, the merging step in the (k+1)th level needs $O[(N/\log N)\cdot 2^k]$ operations. The total running time needed to construct a minimal spanning tree of N vertices is

$$T_p(N) = O(N) + \sum_{k=2}^{\log\log N} \left(\frac{N}{\log N}\right)\cdot 2^{k-1}$$
$$= O(N) + O(N)$$
$$= O(N)$$

where the first term in above equation is the time spent in the processors of the first level.

Therefore, in the worst case, the parallel complexity of MINISPAN is O(N) with O(logN) processors. It is appropriate to point out that our method of organizing data is similar to those used in [Bentley 1980] and [Gowda, Kirkpatrick, Lee and Naamad 1983].

SECTION 5. CONCLUDING REMARKS

In this paper, we have presented an algorithm to construct minimal spanning trees. This algorithm is based the concept of Voronoi diagrams. The input of this algorithm is a set of 2-dimensional points. It is easy to comprehend and can be hand simulated, avoids unnecessary distance calculations, can be implemented as a parallel program and can construct minimal spanning trees in linear time complexity using O(logN) processors. It is different from most minimal spanning trees algorithms, except those in [Bentley and Friedman 1978, Bentley 1980, Shamos 1978] because our algorithm uses the geometrical properties of the input data. It uses many ideas in [Shamos 1978]. Shamos [1978] suggests to construct a minimal spanning tree by extracting it from the Voronoi diagram (actually the Delaunay triangulation). Our algorithm is more direct. It constructs a minimal spanning tree straightforwardly. It does not have to construct a Voronoi diagram because the divide-and-conquer method of constructing Voronoi diagrams can be used to construct minimal spanning trees. Our algorithm is in the same spirit of [Bentley and Friedman 1978] as we try to avoid unnecessary distance calculations.

We like to point out here that by fully utilizing the property of two-dimensional points, we have obtained a parallel algorithm whose complexity is O(N), and uses O(logN) processors. In Table 5-1, we have given a comparison of all of the parallel spanning tree algorithms. We by no means imply that our method is superior to others. We merely imply that if data are two-dimensional points, then a more efficient algorithm can be obtained.

We are now in the process of extending our result to higher dimensional space [Yao 1982]. We are also investigating the average performance of MINISPAN. The material in [Bentley, Weide and Yao 1980] and [Weide 1978] are quite useful.

REFERENCES

[1] Aho, A. V., Hopcroft, J. E. and Ullman, J. D., The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, Mass., 1974.

[2] Bentley, J. L. (1980): A parallel algorithm for construction minimum spanning trees, Journal of Algorithms, Vol. 1, No. 1, 1980, pp. 51-59.

[3] Bentley, J. L. and Friedman, J. H. (1978): Fast algorithms for constructing minimal spanning trees in coordinate spaces, IEEE Transactions on Computers, Vol. C-27, No. 2, 1978, pp. 97-105.

[4] Bentley, J. L. and Kung, H. T. (1979): A tree machine for searching problems, Proceedings, IEEE 1979 International Conference on Parallel Processing, 1979, pp. 257-266.

[5] Bentley, J. L., Weide, B. W. and Yao, A. C. C. (1980): Optimal expected

time analysis for closest point problem, ACM Transactions on Mathematical Software, Vol. 6, No. 4, Dec. 1980, pp. 563-580.

[6] Blum, M., Floyd, R. W., Pratt, V. R., Rivest, R. L. and Tarjan, R. E. (1972): Time bound for selection, Journal of Computer and System Sciences, Vol. 7, No. 4, 1972, pp. 724-742.

[7] Burkhard, W. A. and Keller, R. M. (1973): Some approaches to best-match files searching, Communications of Association for Computing Machinery, Vol. 16, No. 4, 1973, pp. 230-236.

[8] Cheriton, D. and Tarjan, R. E., (1976): Finding minimum spanning trees, SIAM Journal on Computing, Vol. 5, No. 4, 1976, pp. 724-742.

[9] Chin, F. Y., Lam, J. and Chen, I-N. (1982): Efficient parallel algorithms for some graph problems, Communications of Association for Computing Machinery, Vol. 25, No. 5, 1982, pp. 659-665.

[10] Delaunay, B., (1934): Sur la sphere vide, Bull. Acad. Sci. USSR (VII), Classe Sci. Mat. Nat., 1934, pp. 793-800.

[11] Deo, N. and Yoo, Y. B., (1981): Parallel algorithms for minimal spanning trees problems, Proceedings, IEEE 1981 International Conference on Parallel Processing, 1981, pp. 188-189.

[12] Dijkstra, E. W., (1959): A note on two problems with graphs, Numerical Mathematics, Vol. 1, 1959, pp. 269-271.

[13] Friedman, J. H., Baskett, F. and Shustek, J., (1975): An algorithm for finding nearest neighbors, IEEE Transactions on Computers, Vol. C-24, 1975, pp. 1000-1006.

[14] Friedman, J. L., Bentley, J. L. and Finkel, R. A., (1977): An algorithm for finding best matches in logarithmic expected time, ACM Transactions on Mathematical Software, Vol. 3, 1977, pp. 209-226.

[15] Fukunaga, K. and Narenda, P. M., (1975): A branch and bound algorithm for computing K-nearest neighbors, IEEE Transactions on Computers, Vol. C-24, 1975, pp. 750-753.

[16] Gowda, I. G., Kirkpatrick, D. G., Lee, D. T. and Naamad, A., (1983): Dynamic Voronoi diagrams, to appear in IEEE Transactions on Information Theory.

[17] Gower, J. C. and Ross, J. S., (1969): Minimal spanning trees and single linkage cluster analysis, Applied Statistics, Vol. 18, 1969, pp. 54-64.

[18] Hodder, I. and Orton, C., (1976): Spatial Analysis in Archaeology, Cambridge, 1976.

[19] Johnson, S. C. (1967): Hierachical clustering schemes, Psychometrika, Vol. 32, Sept. 1967, pp. 241-254.

[20] Kang, A. N. C., Lee, R. C. T., Chang, C. L. and Chang, S. K., (1977): Storage reduction through minimal spanning trees and spanning forests, IEEE Transactions on Computers, Vol. C-26, 1977, pp. 425-434.

[21] Kruskal, J. B. J., (1956): On the shortest spanning subtree of a graph and the traveling salesman problem, Proceedings of American Mathematics Society, No. 7, 1956, pp. 48-50.

[22] Lee, D. T. and Schachter, B. J., (1980): Two algorithms for constructing Delaunay triangulations, International Journal of Computer and Information Sciences, Vol. 9, No. 3, 1980, pp. 219-242.

[23] Lee, R. C. T., (1981): Clustering analysis and its applications, in Advances in Information Systems Science, (edited by J. T. Tou), Plenum Press, New York, N. Y., 1981, pp. 169-292.

[24] Lee, R. C. T., Chin, Y. H. and S. C. Chang, (1976): Application of principal component analysis to multikey searching, IEEE Transactions on Software Engineering, Vol. SE-2, No. 3, 1976, pp. 185-193.

[25] Lee, R. C. T., Slagle, J. R. and Blum H., A triangulation method for the sequential mapping of points from N-space to 2-space, IEEE Transactions on Computers, Vol. C-26, No. 3, 1976, pp. 310-313.

[26] Leob, A. L., (1976): Space Structure, Addison-Wesley, Reading, Mass., 1976.

[27] Loberman, H. and Weinberger, A., (1957): Formal procedures for connect-

ing terminals with a minimum total wire length, Vol. 4, Oct. 1957, pp. 428-437.

[28] Marden, M., (1966): Geometry of Polynomials, American Mathematical Society, 1966.

[29] Nath, D. and Maheshwar, S. N., (1982): Parallel algorithms for the connected components and minimal spanning tree problems, Information Processing Letters, Vol. 14, No. 1, 1982, pp. 7-11.

[30] Ottman, T. A., Rosenburg, A. L. and Stockmeyer, L. J., (1982): A dictionary machine (for VLSI), IEEE Transactions on Computers, Vol. C-32, No. 9, 1982, pp. 892-897.

[31] Prim, R. C., (1957): Shortest connecting networks and some generalizations, Bell System Technical Journal, Vol. 36, 1957, pp. 1389-1401.

[32] Rogers, C. A., (1964): Packing and Covering, Cambridge University Press, Cambridge, 1964.

[33] Savage, C. and Ja' Ja', J., (1981): Fast efficient parallel algorithms for some graph problems, SIAM Journal on Computing, Vol. 10, No. 4, 1981, pp. 682-691.

[34] Shamos, M. I., (1978): Computational Geometry, Ph.D. Thesis, Yale University.

[35] Shamos, M. I. and Hoey, D., (1975): Closest point problems, 16th Annual IEEE Symp. on Foundations of Computer Science, 1975, pp. 151-162.

[36] Shen, C. W. and Lee, R. C. T., (1977): A nearest neighbor search technique with short zero in time, Proceedings of 1978 International Computer Software and Applications Conference, IEEE, 1978, pp. 470-475.

[37] Sollin, M., (1977): An algorithm attributed to Sollin, in Introduction to the Design and Analysis of Algorithms, Goodman, S. E. and Hedetniemi, S. T., Section 5.5, McGraw-Hill, N. Y., 1977.

[38] Tarjan, R. E.,(1982): Sensitivity analysis of minimal spanning trees and shortest path trees, Information Processing Letters, Vol. 14, No. 1, 1982, pp. 30-33.

[39] Voronoi, G., (1908): Nouvelles applications des parametres continus a la theole des formes quadratiques, Deuxieme Menoire: Recherches sur les parallelloedres, Deuxieme angew. Math., Vol. 134, 1908, pp. 198-287.

[40] Weide, B. W., (1978): Statistical methods in algorithm design and analysis, Ph.D. Dissertation, Carnegie-Mellon University.

[41] Yao, A. C., (1975): An O(EloglogV) algorithm for finding minimum spanning trees, Information Processing Letters, Vol. 4, 1975, pp. 21-23.

[42] Yao, A. C., (1982): On constructing minimal spanning trees in k-dimensional space and related problems, SIAM Journal on Computing, Vol. 11, No. 4, 1982, pp. 721-736.

[43] Zahn, C. T., (1971): Graph-theoretical methods for detecting and describing Gestalt clusters, IEEE Transactions on Computers, Vol. C-20, No. 1, 1971, pp. 68-86.

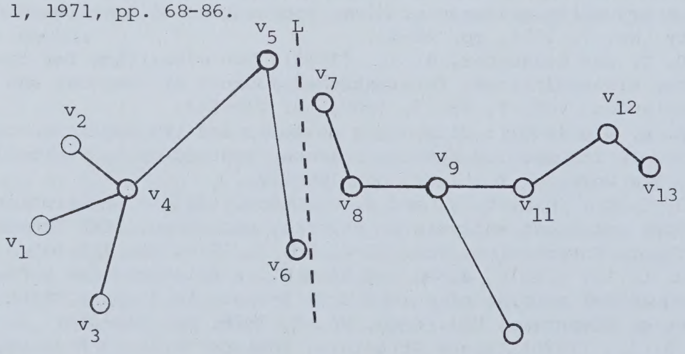


Fig. 2-1

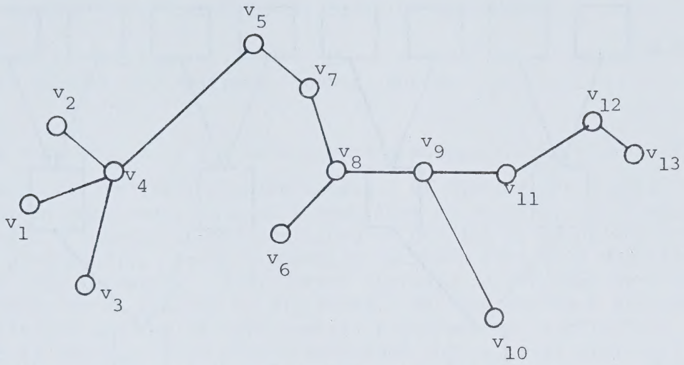

v_5
v_7
v_2
v_{12}
v_8
v_9
v_{13}
v_4
v_{11}
v_1
v_6
v_3
v_{10}

Fig. 2-2

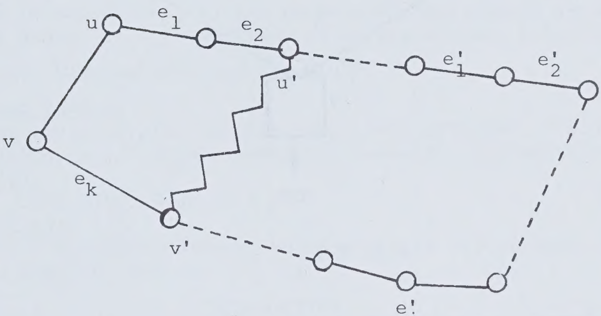

u
e_1
e_2
e'_1
e'_2
u'
v
e_k
v'
e'_j

Fig. 3-1

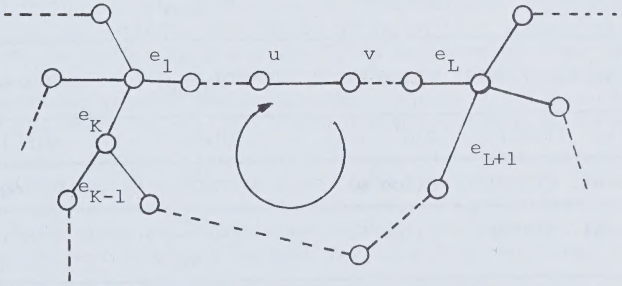

e_1
u
v
e_L
e_K
e_{L+1}
e_{K-1}

Fig. 4-1

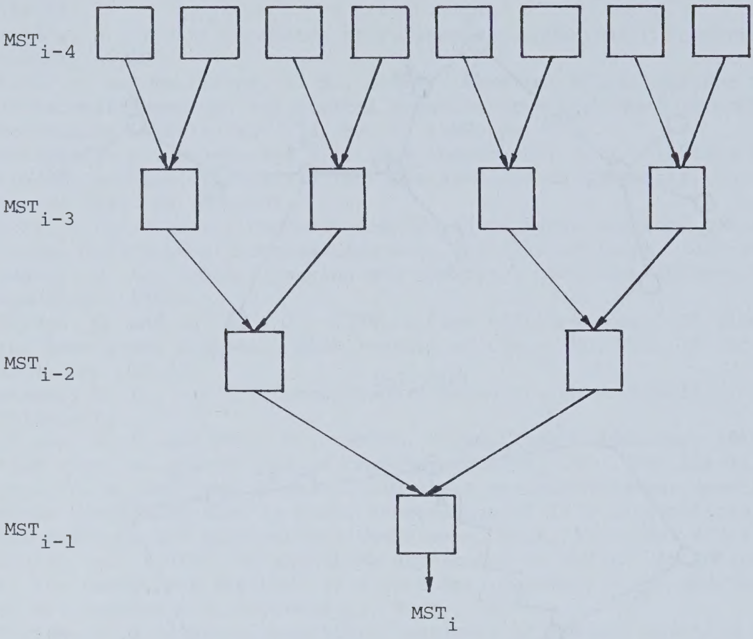


Fig. 4-2

Algorithms	T=Parallel Complexity	P=No. of Processors	T x P
Sollin [1977]	$O(N\log N)$	$O(N)$	$O(N^2\log N)$
Bentley [1980]	$O(N\log N)$	$O(\frac{N}{\log N})$	$O(N^2)$
Savage et al. [1981]	$O(\log^2 N)$	$O(\frac{N^2}{\log N})$	$O(N^2\log N)$
Deo et al. [1981]	$O(N^{1.5})$	$O(\sqrt{N})$	$O(N^2)$
Nath et al. [1982]	$O(\log^2 N)$	$O(N^2)$	$O((N\log N)^2)$
Chin et al. [1982]	$O(\log^2 N)$	$O(\frac{N^2}{\log^2 N})$	(N^2)
Our Method	$O(N)$	$O(\log N)$	$O(N\log N)$

Table 5-1

COMPUTING LARGEST EMPTY CIRCLES WITH LOCATION CONSTRAINTS

Godfried T. Toussaint
McGill University, Montreal, Quebec, Canada

Let $Q = \{q_1, q_2, \ldots, q_n\}$ be a set of n points on the plane. The largest empty circle (LEC) problem consists of finding the largest circle C with center in the convex hull of Q such that no point $q_i \in Q$ lies in the interior of C. Shamos recently outlined an O(n log n) algorithm for solving this problem [1]. In this paper it is shown that this algorithm does not always work correctly. A different approach is proposed here and shown to also result in an O(n log n) algorithm. The new approach has the advantage that it can also solve more general problems. In particular, it is shown that if the center of C is constrained to lie in an arbitrary convex n-gon an O(n log n) algorithm can still be obtained. Finally, an O(n log n + k log n) algorithm is given for solving this problem when the center of C is constrained to lie in an arbitrary simple n-gon P, where k denotes the number of intersections occurring between edges of P and edges of the Voronoi diagram of Q and $k \leq O(n^2)$.

Key Words and Phrases:
largest empty circle, facility location, polygons, Voronoi diagram, algorithms, complexity, operations research, computational geometry.
CR Categories:
3.36, 3.63, 5.25, 5.32, 5.5

Author's Address:
School of Computer Science, 805 Sherbrooke Street West, Montreal,
Quebec H3A 2K6, Canada.

INTRODUCTION

Let $Q = \{q_1, q_2, \ldots, q_n\}$ be a set of n points on the plane and let CH(Q) denote the convex hull of Q. The largest empty circle (LEC) problem consists of finding the largest circle the center of which lies in CH(Q) such that no point of Q lies in the interior of the circle. It is clear that the location of the center must be constrained. Otherwise we simply set the center at infinity and obtain an infinitely large circle. It is assumed that the points in Q are specified in terms of their cartesian coordinates and they lie in general position in the sense that no three are collinear and no four are cocircular. Figure 1 illustrates a set of points and the solution circle. Until recently, the best known algorithm for solving this problem required $O(n^3)$ time [1]. More recently, Shamos[2] outlined an O(n log n) algorithm based upon computing the Voronoi diagram of Q, V(Q). It is assumed here that the reader is familiar with the Voronoi diagram properties discussed in [2]. For example, the Voronoi diagram has at most m = 2n-4 Voronoi vertices (V - vertices) $V = \{v_1, v_2, \ldots, v_m\}$ and each v_i is the circumcenter of three points of Q determined by a Delaunay triangle in the dual of the Voronoi diagram of Q. Furthermore these circles are empty. If all v_i in V lie in CH(Q) then one of these vertices is the center of the solution circle. However, not all the V-vertices of V(Q) need lie in CH(Q) in which case the solution circle may be a V-vertex or it may lie at the intersection point of an edge of V(Q) and an edge of CH(Q) as will be shown in section 2.2. Figure 1 illustrates the Voronoi diagram of a set of points where the solution circle is centered on a V-vertex. Figure 2 illustrates a situation where

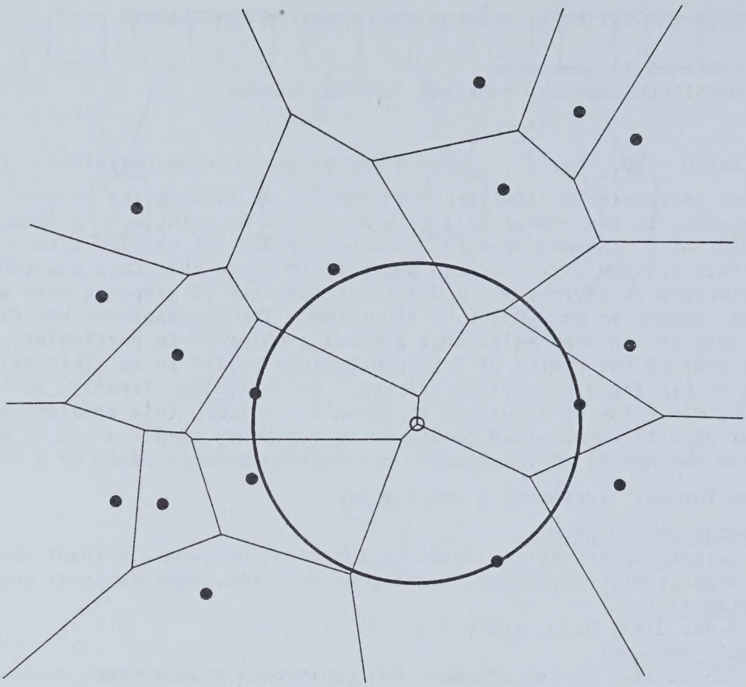


Figure 1: The Voronoi diagram of a set of points such that the largest empty circle is centered on a Voronoi vertex.

the solution center is not a V-vertex. Shamos' algorithm consists of two parts: the first part computes the Voronoi diagram of Q and the magnitudes of all the circles centered at V-vertices that lie in CH(Q), and the second part looks for intersection points of V-edges with CH(Q) edges and computes the magnitudes of the empty circles centered at these intersection points, (see Figure 2). Now each CH-edge such as q_iq_j in Figure 2 determines an unbounded V-edge r_1 collinear with the perpendicular bisector of the CH edge. In [2], Shamos claims that each CH-edge such as q_iq_j either intersects r_1 or the two V-edges adjacent to r_1 (r_2 and r_3 in Figure 2), i.e., that a CH-edge intersects at most two V-edges. Drawing on this, the second part of the algorithm checks each CH-edge with the corresponding one or three (as the case may be) V-edges for intersection points and computes the corresponding empty circles. Since the Voronoi diagram can be computed in O(n log n) time and for each CH-edge the circles are computed in O(1) time the algorithm has a total running time of O(n log n).

Unfortunately, the above claim is false and such an algorithm may miss some circles; indeed it may miss the largest empty circle itself and thus fail. One counterexample to the above algorithm is illustrated in Figure 3. Note that *all* V-vertices lie outside CH(Q), r_2 and r_3 do not intersect the corresponding CH-edge, and indeed a CH-edge may intersect more than two V-edges; in fact, it may intersect O(n) V-edges. In Figure 3, the LEC (solid curve) is not detected and instead the algorithm exits with a suboptimal solution (dashed curve). These examples reopen the question of whether an O(n log n) upper bound exists for this problem. Figure 3 suggests that we may be able to efficiently search the Voronoi diagram starting at every unbounded V-edge until all intersections of V-edges with the corresponding CH-edge are found. In fact, in two earlier works Shamos [3], [4] suggests a depth-first and breadth-first search, respectively, for solving this problem. However, it is not clear what the complexity of such a step would be. Although in [3] and [4] Shamos claims this step can be done in O(n) time, no proofs are given and the simplistic arguments are not convincing. For example, in [3] it is claimed that each edge of V(Q) can intersect at most one edge of CH(Q) but, as Figure 3 illustrates, this is not necessarily true. Also in [4] it is claimed that in a breadth-first search no V edge will be examined more than once. However, Figure 4 illustrates a situation where breadth-first search examines some V-edges twice. Rather than try to prove the correctness of the approaches taken in [3] and [4], in this paper we propose a new approach to this problem which not only yields an algorithm with O(n log n) running time, but the algorithm is simpler, admits an easy proof of correctness, and extends to more interesting and applicable generalizations of the LEC problem.

One interpretation of the LEC problem is to let Q denote a set of cities in a country. If it is desired to place an installation such as a nuclear power plant in a location that maximizes the distance from the power plant to the nearest city then the location is the center of the largest empty circle. However, the convex hull of the cities is not a very realistic constraint on the location of a nuclear power plant. The CH(Q) may intersect other countries and the resulting solution may call for placing the plant in a country that may not grant permission. Alternately, the country in which the cities lie may have a large peninsula protruding from CH(Q). The optimal solution may indeed call for placing the plant on the peninsula and thus constraining the location to CH(Q) would yield a suboptimal solution. Finally, one may simply wish to constrain the location to some region within the country such as a desert. Furthermore, this region may not contain a V-vertex and may not intersect V-edges in which case the CH constraint fails completely. In this

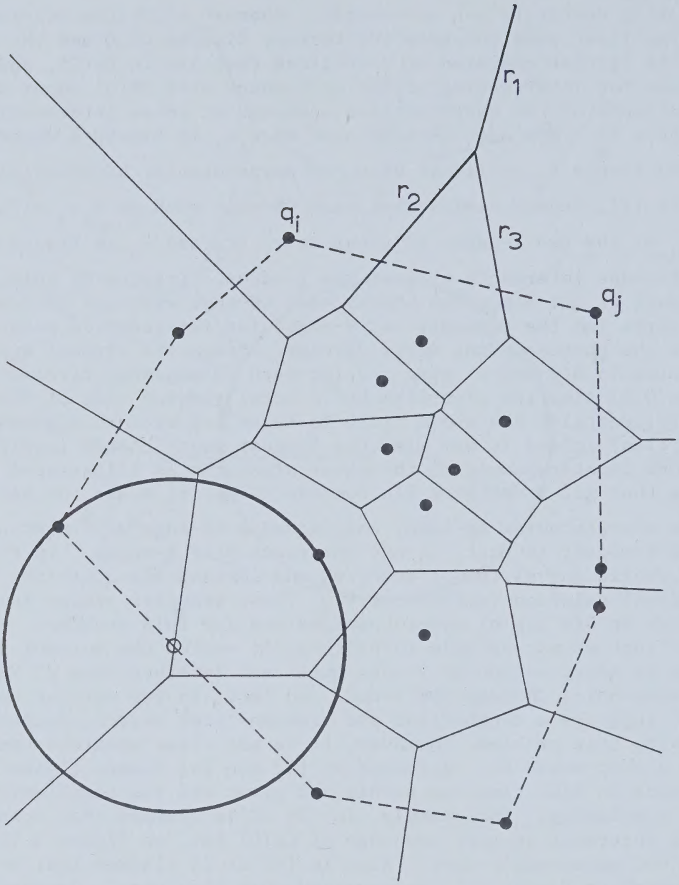


Figure 2: Illustrating the case where the largest empty circle is centered at an intersection point of a Voronoi edge and a convex hull edge.

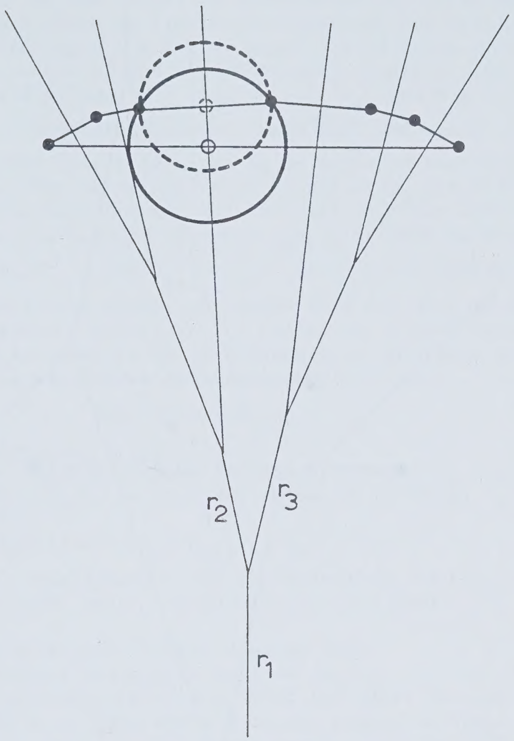


Figure 3: A counter-example to the algorithm of Shamos. All the Voronoi vertices lie outside the convex hull and r_2 and r_3 do not intersect the convex hull edge corresponding to r_1.

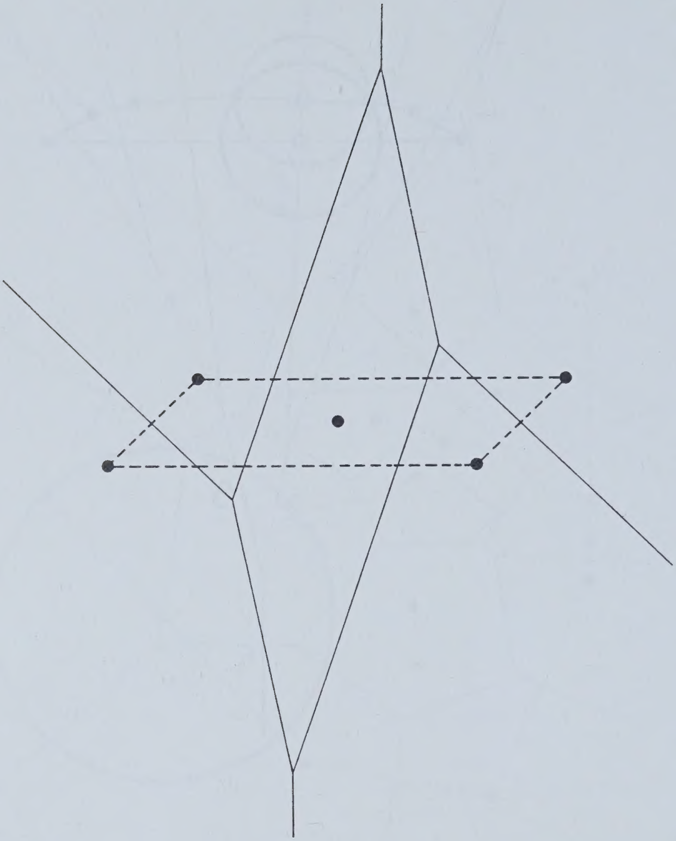


Figure 4: Illustrating a situation in which breadth-first search of the Voronoi diagram examines some Voronoi edges more than once.

paper, it is shown that if the center is constrained to lie in an arbitrary convex polygon with n sides an O(n log n) algorithm can still be obtained. Finally, an O(n log n + k log n) algorithm is given for solving this problem when the center of the circle is constrained to lie in an arbitrary simple n-sided polygon P = (p_1, p_2,..., p_n) where k denotes the number of intersections occurring between edges of P and edges of V(Q) and $k \leq O(n^2)$. It is assumed that P is given as an array of vertices, in clockwise order, along with their cartesian coordinates, and that is in *standard form*, i.e., the vertices are distinct and no three consecutive vertices are collinear. A pair of vertices $p_i p_{i+1}$ defines an edge of the polygon where i = 1, 2,..., n and $p_{n+1} = p_1$. Clearly the number of vertices of P need not equal the number of points in Q but such an assumption will simplify the complexity notation. If the reader wishes to observe the role played by each, it is straightforward to repeat the complexity analysis when P and Q have different cardinalities.

NEW ALGORITHMS

2.1 The Convex Hull Location Constraint

We first consider the LEC problem treated by Shamos [2] and Dasarathy and White [1] where the center is constrained to lie in CH(Q).

Algorithm LEC - 1

Input: A set of points Q = {q_1, q_2,..., q_n}.

Output: {x, y, r}, the location (x, y coordinates) of the center and the radius r of the largest empty circle with $(x,y) \in$ CH(Q).

```
Begin
    Step 1: Compute the Voronoi diagram V(Q).
    Step 2: Compute the convex hull CH(Q).
    Step 3: Preprocess CH(Q) for point inclusion testing in
            O(log h) time where h is the number of edges in CH(Q).
    Step 4: Determine which Voronoi vertices lie in CH(Q) and
            compute their empty circles.
    Step 5: Determine the intersection points of the Voronoi edges
            and the edges of CH(Q) and compute their empty circles.
    Step 6: Select as the output the largest circle encountered
            in steps 4 and 5.
End
```

Note that the structure of this algorithm is the same as Shamos'. It is in step 5 that we need an efficient algorithm for computing the intersection points. Several such algorithms exist [5], [12], without recourse to breadth-first or depth-first search algorithms of the V-diagram, which lead to the following theorem.

Theorem 1:

Algorithm LEC - 1 computes the largest empty circle in O(n log n) time.

Proof:

The correctness follows from the fact that the solution center lies either at a V-vertex or at an intersection point of a V-edge and a CH-edge, and all the circles corresponding to these centers are tested. Thus it only remains to consider the complexity. Step 1 can be done in O(n log n) time [2]. In step 2, the CH(Q) can be obtained in O(n) time from the Voronoi diagram [2]. Step 3 can be done with sectors in O(h) time and O(h) space [2]. To determine whether a V-vertex lies in CH(Q) in step 4 requires O(log h) time and computing each circle can be done in constant time since a V-vertex points to its three equidistant members of Q which determine the radius r. Since there are at most 2n-4

V-vertices step 4 can be computed in O(n log h) time. Several fast algorithms exist for determining whether a line or line segment intersects a convex polygon and for computing the intersection points if they exist. For example, Chazelle [5] gives a procedure that combines Fibonacci search and binary search to obtain an O(log n) algorithm for computing the intersection of a line segment and a convex n-gon. In step 5 we have at most 3n-6 V-edges and h CH-edges and thus with Chazelle's algorithm computing the intersection points in step 5 can be accomplished in O(n log h) time. Since a line segment intersects a convex polygon at most twice we obtain O(n) intersection points for which to compute empty circles. Furthermore, each intersection point identifies a V-edge which in turn identifies the two equidistant points of Q that determine the radius of the empty circle. Thus all the circles can be computed in O(n) time. Finally, step 6 requires constant time and the total running time of the algorithm is dominated by step 1 and is O(n log n).

With the approach to step 5 given here it is clear that, unlike Shamos' algorithm, we can compute the intersection points of V-edges not just with edges of CH(Q) but with edges of any convex (or non-convex for that matter) polygon. Thus this approach suggests these two generalizations of the LEC problem discussed in the introduction. However, we must be careful to consider the case where the constraint polygon does not intersect any V-edge. In the next section we show how to handle these situations in O(n log n) time for convex polygons and in $O(n^2 \log n)$ time for arbitrary simple polygons.

2.2 The Convex and Simple Polygon Location Constraints

It turns out that if the center of the largest empty circle is constrained to lie in a polygon $P = (p_1, p_2, \ldots, p_n)$ then the solution is one whose center lies either at a V-vertex of V(Q), an intersection point of a V-edge and an edge of P, or at a *vertex* of P. Let $T = \{t_1, t_2, \ldots, t_k\}$ denote the set of intersection points of edges of P with edges of V.

Theorem 2:

Given a set of points $Q = \{q_1, q_2, \ldots, q_n\}$ and a polygon $P = (p_1, p_2, \ldots, p_n)$ the largest circle C with center O such that $O \in P$ and no point in Q lies in the interior of C must be a circle centered at either:

(a) a Voronoi vertex v_i for some i

or (b) an intersection point t_j for some j

or (c) a vertex of P, p_ℓ for some ℓ.

The above theorem suggests the following algorithm for computing the largest empty circle with its center constrained to lie in a polygon P.

Algorithm LEC - 2

Input: A set of points $Q = \{q_1, q_2, \ldots, q_n\}$ and a simple polygon $P = (p_1, p_2, \ldots, p_n)$.

Output: The largest empty circle with center in P.

```
Begin
    Step 1: Compute the Voronoi diagram V(Q)
    Step 2: Preprocess P for point inclusion testing in O(log n) time.
    Step 3: Determine which Voronoi vertices of V(Q) lie in P and compute their empty circles.
    Step 4: Compute $T = \{t_1, t_2, \ldots, t_k\}$, i.e., the intersections
```

```
         of V-edges and P-edges and for each $t_i$ its empty circle.
Step 5:  Preprocess V(Q) for fast nearest neighbor queries.
Step 6:  For each $p_i \in P$ determine the nearest neighbor $q_j \in Q$
         and compute its empty circle.
Step 7:  Select and output the largest circle found in steps
         3, 4, and 6.
End
```

Note that when P is convex step 2 of Algorithm LEC - 2 is the same as step 3 of LEC - 1 where the complexity is now O(log n) rather than O(log h) for point inclusion testing. Thus the only difference between the two algorithms is that we must now compute empty circles for vertices of P, i.e., steps 5 and 6.

Theorem 3:

For convex location-constraint polygons Algorithm LEC - 2 computes the largest empty circle in O(n log n) time.

Proof:

The correctness follows from theorem 2. We thus consider the complexity of steps 5 and 6. V(Q) is a *planar subdivision*. Several algorithms exist for searching location in a planar subdivision in O(log n) time [7] - [10]. In particular Kirkpatrick [10] gives an easily implementable algorithm that requires only O(n) storage and O(n log n) preprocessing time. Thus step 5 can be done in O(n log n) time. Step 6 now involves locating each vertex p_i of P, to determine the nearest $q_j \in Q$ to p_i, in the Voronoi diagram V(Q). Since there are n vertices and each query takes O(log n) time, step 6 requires O(n log n) time. Thus the entire algorithm runs in O(n log n) time. Q.E.D.

Theorem 4:

For simple location-constraint polygons Algorithm LEC - 2 computes the largest empty circle in $O(n^2 \log n)$ time.

Proof:

Again, the correctness follows from theorem 2 and we consider the complexity. Whether P is convex or not only affects steps 2 and 4 in LEC - 2. Consider step 2. P is clearly a planar subdivision and although we cannot use the simple linear space and time algorithm based on sectors we can use the procedures in [10]. Thus the complexity of step 2 is now O(n log n). We turn to step 4. We must compute $T = \{t_1, t_2, \ldots, t_k\}$. For convex polygons a V-edge can have at most two intersections and since there are at most 3n-6 V-edges k = O(n). With simple polygons, however, each V-edge may intersect O(n) P-edges and thus in the worst case $k = O(n^2)$. To compute T we must solve a classical line segment intersection problem where we are given O(n) V and P edges and we must find the k intersection points. Bentley and Ottmann [11] have shown that this can be done in O(n log n + k log n) time. Thus the total complexity is dominated by step 4 and is O(n log n + k log n). In the worst case this reduces to $O(n^2 \log n)$ time. Q.E.D.

REFERENCES

[1] B. Dasarathy and L. J. White, "On some maximin location and classifier problems", Computer Science Conference, Washington, D.C., 1975 (unpublished lecture).

[2] M. I. Shamos, "Computational geometry", Ph.D. thesis, Yale University 1978.

[3] M. I. Shamos, *Problems in Computational Geometry*, Carnegie-Mellon

University, 1977.

[4] M. I. Shamos and D. Hoey, "Closest-point problems", *Sixteenth Annual IEEE Symposium on Foundations of Computer Science,* October 1975, pp. 151-162.

[5] B. M. Chazelle, "Computational geometry and convexity", Ph.D. thesis, Carnegie-Mellon University, July 1980.

[6] G. T. Toussaint and B. K. Bhattacharya, "On geometric algorithms that use the furthest-point Voronoi diagram", Tech. Rept. No. SOCS-81,3, School of Computer Science, McGill University, January 1981.

[7] R. J. Lipton and R. E. Tarjan, "Applications of a planar separator theorem", *SIAM J. on Computing,* Vol. 9, No. 3, August 1980, pp. 615-627.

[8] D. T. Lee and F. P. Preparata, "Location of a point in a planar subdivision and its applications", *SIAM J. on Computing,* Vol. 8, 1979, pp. 542-545.

[9] F. P. Preparata, "A note on locating a set of points in a planar subdivision", *SIAM J. Computing,* Vol. 8, 1979, pp. 542-545.

[10] D. Kirkpatrick, "Optimal search in planar subdivisions", *SIAM J. Computing,* Vol. 12, No. 1, February 1983, pp. 28-35.

[11] J. L. Bentley and T. A. Ottmann, "Algorithms for reporting and counting geometric intersections", *IEEE Trans. Computers,* Vol. C-28, September 1979, pp. 643-647.

[12] B. Chazelle and D. Dobkin, "Detection is easier than computation", *ACM Symposium on Theory of Computing,* Los Angeles, CA., May 1980, pp. 146-153.

A FURTHER INVESTIGATION OF THE LLOYD-MAX ALGORITHM FOR QUANTIZER DESIGN

FU-SHENG LU
Department of Electrical Engineering
University of Texas at Austin
Austin, Texas 78712

and

GARY L. WISE
Departments of Electrical Engineering and Mathematics
University of Texas at Austin
Austin, Texas 78712

ABSTRACT

The Lloyd-Max algorithm is currently one of the most popular algorithms for the design of scalar minimum mean squared error quantizers. However, two tactical problems exist with this iterative scheme: initializing the algorithm and updating the algorithm. This paper presents a straightforward method of initializing the algorithm for several distributions on the input. Also, a simple and efficient update procedure is given for a wide class of input distributions.

I. INTRODUCTION

Lloyd [1] and Max [2] are generally credited with pioneering results in the design of scalar minimum mean squared error quantizers. They independently found necessary conditions for optimality, and they introduced an iterative scheme for quantizer design. This iterative method is commonly known as the Lloyd-Max algorithm or Max's algorithm. Following the efforts of Lloyd and Max, many researchers have computed optimum quantizer parameters for several other distributions (see, for example, [3-6]). The Lloyd-Max algorithm is very simple to program into a digital computer, and it is popularly adopted by many researchers.

Unfortunately, two potential problems arise with the Lloyd-Max scheme. The first problem is how to get a good initial guess for starting the iterative scheme. The second problem is how to update the algorithm; that is, how to modify the key parameter intelligently during the computation so as to reduce the number of necessary iterations. In this paper, we propose simple and efficient solutions to these problems.

II. A SIMPLE METHOD OF OBTAINING A GOOD INITIAL GUESS

Suppose that the input signal has a continuous density $f(\cdot)$ such that there exists an open interval (x',x'') over which $f(\cdot)$ is strictly positive and $\int_{x'}^{x''} f(x)dx = 1$. This open interval could be the real line, for instance. Lloyd [1] and Max [2] showed that an N level minimum mean squared error quantizer must be such that the following equations are satisfied:

$$\int_{x_i}^{x_{i+1}} (t-y_i)f(t)dt = 0 \qquad i=1,\dots,N \tag{1}$$

and

$$x_i = (y_i + y_{i-1})/2 \qquad i=2,\dots,N \tag{2}$$

where $x' = x_1 < x_2 < \dots < x_{N+1} = x''$ denote the interval breakpoints and $y_1 < y_2 < \dots < y_N$ denote the output levels of the N-level quantizer. The first step of the Lloyd-Max procedure is to choose a value for y_1 with which to solve (1) for the value x_2. From (2) we have $y_2 = 2x_2 - y_1$. Then by (1) again we find x_3, and so on. The last integral over (x_N, x_{N+1}) can be used to check the accuracy of the initial guess for y_1. If the last integral is zero within a tolerable error, we use the computed parameters to specify the quantizer; if not, we adjust the value of y_1 and begin the procedure again. Alternatively, instead of y_1, we can first make a guess for y_N, solve (1) for x_N, determine y_{N-1} from (2), solve (1) for x_{N-1}, and so on; and the last integral over (x_1, x_2) can be used to check the accuracy of the initial guess for y_N. Hence, y_1, or y_N, is called the "key parameter" here. If the optimum quantizer is symmetric, we just need to compute one half of the parameters.

This iterative method is simple but Lloyd and Max did not mention how to choose the initial guess for y_1 nor how to modify the value of y_1 during the computation. Several investigators have mentioned that the convergence properties of the Lloyd-Max method are greatly dependent on the initial guess for y_1. They tried some methods for choosing the iteration's initial condition as accurately as possible. For example, Bucklew and Gallagher [7] propose the λ-approximation and the g-approximation methods and find the g-approximation is a better approach. The λ-approximation method is based on a concept known as the asymptotic level density $\lambda(x)$ for the minimum mean square error quantizer. The g-approximation method uses the companding function to work backwards from the known uniform quantizer over [0,1] in order to estimate the initial output level. In the companding model, the N-level uniform quantizer on [0,1] has output levels

$$z_i = (2i-1)/2N \qquad i=1,\dots,N.$$

Hence, the g-approximation results in the approximation

$$\hat{y}_i = g^{-1}(z_i) = g^{-1}\left(\frac{2i-1}{2N}\right), \tag{3}$$

where $g^{-1}(x)$ is the expansion nonlinearity. Assume that the input has a density function $f(\cdot)$ such that $\int_{-\infty}^{\infty} [f(x)]^{1/3}\, dx < \infty$. For minimum mean square error quantization, the asymptotically optimum compressor is given by

$$g(x) = \int_{-\infty}^{x} [f(t)]^{1/3}\, dt \Big/ \int_{-\infty}^{\infty} [f(t)]^{1/3}\, dt. \tag{4}$$

In fact, $g(x)$ is seldom an elementary function, so the g-approximation is not an easy method. Here we suggest a simple approximation which we call the s-approximation. This results in a much easier and more accurate method.

Figure 1 shows, from $N = 4$ to 64, the relationship of the tabulated N^{th} output level y_N^* of a minimum mean squared error quantizer with $\log_e N$ for the Gaussian, Laplace, and Rayleigh density functions, respectively, where

N denotes the number of output levels. These density functions are discussed in [7]. The nearly straight line behavior suggests an approximation for y_N^* as

$$\hat{y}_N = a \log_e N + b, \tag{5}$$

and we refer to this as the s-approximation method. The parameters a and b are determined by a classical least squares method. For the Gaussian and Laplace densities, the optimum quantizers are symmetric [8]; so that $y_1 = -y_N$. For the Rayleigh density, we found that a plot of y_1^* versus $\log_e N$ was not as well approximated by a straight line as a plot of y_N^* versus $\log_e N$. For this reason we will confine our interest in this section to obtaining a good initial guess of y_N. Alternatively, we could replace f(x) by f(-x) and note that the "first" level for f(-x) is the negative of the "last" level for f(x).

The zero mean Gaussian density is given by

$$f_G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-x^2/2\sigma^2} .$$

In this case the asymptotically optimum expansion nonlinearity is

$$g_G^{-1}(z) = \begin{cases} (\sqrt{6}\ \sigma)\ \mathrm{erf}^{-1}(2z-1) & 1/2 \leq z \leq 1 \\ (-\sqrt{6}\ \sigma)\ \mathrm{erf}^{-1}(1-2z) & 0 \leq z < 1/2 \end{cases}$$

where $\mathrm{erf}^{-1}(x)$ is the inverse error function. Using the s-approximation method, for $\sigma = 1$, we have

$$\hat{y}_{N,s} \approx 0.77848 \log_e N + 0.54448$$

as the estimate of the N^{th} output level. If we use the g-approximation method for the zero mean and unit variance Gaussian signal, then we get

$$\hat{y}_{N,g} = \sqrt{6}\ \mathrm{erf}^{-1}\ (1 - \frac{1}{N})$$

as the estimate of the N^{th} output level.

Adopting similar performance measures as Bucklew and Gallagher [7], we define

$$E_s \triangleq |(\hat{y}_{N,s} - y_N^*) / y_N^*|$$

and

$$E_g \triangleq |(\hat{y}_{N,g} - y_N^*) / y_N^*| .$$

The relationship of E_g and E_s versus values of N from 4 to 64 is shown in Figure 2. We see that E_s is much smaller than E_g for all these values of N.

The Laplace density function

$$f_L(x) = \frac{\gamma}{2} e^{-\gamma|x|}$$

is frequently studied in speech signal processing. The asymptotically optimum expansion nonlinearity is

$$g_L^{-1}(z) = \begin{cases} \frac{3}{\gamma} \log_e \frac{1}{2(1-z)} & \frac{1}{2} \leq z < 1 \\ \frac{3}{\gamma} \log_e 2z & 0 < z < \frac{1}{2}. \end{cases}$$

For $\gamma = \sqrt{2}$ (i.e. unit variance), by the g-approximation we have

$$\hat{y}_{N,g} = \frac{3}{\sqrt{2}} \log_e N \approx 2.12132 \log_e N.$$

With the s-approximation method, we get

$$\hat{y}_{N,s} \approx 1.99303 \log_e N - 1.057.$$

In Figure 3, we plot E_g and E_s for values of N from 4 to 64 for the Laplace density. Again, the s-approximation is much better than the g-approximation for all these values of N.

For the Rayleigh density function

$$f_R(x) = \begin{cases} \frac{x}{\sigma^2} e^{-x^2/2\sigma^2} & , x \geq 0 \\ 0 & , \text{otherwise,} \end{cases}$$

the asymptotically optimum expansion nonlinearity is

$$g_R^{-1}(z) = \sigma[6B^{-1}(z\Gamma(\tfrac{2}{3}))]^{1/2} \qquad 0 \leq z \leq 1,$$

where $B^{-1}(\cdot)$ is the inverse function of B(s) with

$$B(s) \triangleq \int_0^s \exp(-u)\, u^{-1/3}\, du,$$

and $\Gamma(x)$ is the gamma function. For $\sigma=1$, with the s-approximation,

$$\hat{y}_{N,s} \approx 0.69082 \log_e N + 1.65864,$$

and by the g-approximation, we have

$$\hat{y}_{N,g} = \sqrt{6B^{-1}\left(\frac{2N-1}{2N}\, \Gamma(\tfrac{2}{3})\right)}.$$

The graphs E_g and E_s versus values of N from 4 to 64 for the Rayleigh distribution are shown in Figure 4. We can see the s-approximation is still an easier and better method.

III. THE MODIFICATION OF THE KEY PARAMETER DURING THE ITERATION

In this section we assume that the algorithm begins with a guess for y_1 and we discuss an efficient method for modifying y_1 during the computation. This method results in reducing the number of iterations necessary to get the optimum quantizer parameters.

Let

$$\varepsilon \triangleq \int_{x_N}^{x_{N+1}} (t-y_N)f(t)dt = \int_{x_N}^{x''} (t-y_N)f(t)dt. \qquad (6)$$

Notice that ε is a function of the initial output level y_1. Thus the problem of optimum quantization can be posed as the problem of finding a numerical solution, through the iterative procedure, for the nonlinear equation $\varepsilon(y_1) = 0$. We note that a solution to this equation always exists [9]. Also, we note from the inverse mapping theorem [10] that under mild conditions,

$$\frac{\partial y}{\partial x} = \left(\frac{\partial x}{\partial y}\right)^{-1} .$$

In this paper we will henceforth use this property and we will be implicitly assuming that it holds. From (1), with i=1, we have

$$\frac{\partial x_2}{\partial y_1} = \frac{\int_a^{x_2} f(t)dt}{(x_2-y_1)f(x_2)} .$$

For $2 \leq i \leq N-1$ in (2), we have

$$\frac{\partial y_i}{\partial x_i} = 2 - \frac{1}{\frac{\partial x_i}{\partial y_{i-1}}} .$$

Similarly, for $2 \leq i \leq N-1$,

$$\frac{\partial x_{i+1}}{\partial y_i} = \frac{(x_i-y_i)f(x_i) + (\frac{\partial y_i}{\partial x_i})\int_{x_i}^{x_{i+1}} f(t)dt}{(x_{i+1}-y_i)f(x_{i+1})(\frac{\partial y_i}{\partial x_i})} .$$

Now

$$\frac{\partial \varepsilon}{\partial y_1} = \frac{\partial \varepsilon}{\partial y_{N-1}} \prod_{i=1}^{N-2} (\frac{\partial y_{N-i}}{\partial x_{N-i}})(\frac{\partial x_{N-i}}{\partial y_{N-i-1}}) ,$$

and

$$\frac{\partial \varepsilon}{\partial y_{N-1}} = \left[(x_N-y_{N-1})f(x_N)-2\int_{x_N}^{x''} f(t)dt\right](\frac{\partial x_N}{\partial y_{N-1}}) + \int_{x_N}^{x''} f(t)dt.$$

Thus we have

$$\varepsilon'(y_1) \overset{\Delta}{=} \frac{\partial \varepsilon}{\partial y_1} = \left\{\left[(x_N-y_{N-1})f(x_N)-2\int_{x_N}^{x''} f(t)dt\right](\frac{\partial x_N}{\partial y_{N-1}}) + \int_{x_N}^{x''} f(t)dt\right\}$$

$$\cdot \prod_{i=1}^{N-2} (\frac{\partial y_{N-i}}{\partial x_{N-i}})(\frac{\partial x_{N-i}}{\partial y_{N-i-1}}) \qquad (7)$$

for $N > 2$.

Using the Newton-Raphson method, we can modify the value of y_1 during the iterative computations by

$$y_{1,k+1} = y_{1,k} - \frac{\varepsilon(y_{1,k})}{\varepsilon'(y_{1,k})} . \qquad (8)$$

If the optimum quantizer is symmetric about zero, we need only to compute one half of the parameters. In this case, for even N, let $N = 2M$ and define ε_1 as

$$\varepsilon_1 \triangleq \int_{x_M}^{0} (t-y_M)\, f(t)dt. \qquad (9)$$

This yields

$$\varepsilon_1'(y_1) \triangleq \frac{\partial \varepsilon_1}{\partial y_1} = \left\{ \left[(x_M - y_{M-1}) f(x_M) - 2\int_{x_M}^{0} f(t)dt \right] \left(\frac{\partial x_M}{\partial y_{M-1}} \right) + \int_{x_M}^{0} f(t)dt \right\}$$

$$\cdot \prod_{i=1}^{M-2} \left(\frac{\partial y_{M-i}}{\partial x_{M-i}} \right) \left(\frac{\partial x_{M-i}}{\partial y_{M-i-1}} \right) \qquad (10)$$

for $M > 2$. Also, if N is odd, let $N = 2M+1$, and define

$$\varepsilon_2 \triangleq \int_{x_M}^{x_{M+1}} (t-y_M) f(t)dt = \int_{x_M}^{y_M/2} (t-y_M) f(t)dt, \qquad (11)$$

since $x_{M+1} = y_M/2$. In this case

$$\varepsilon_2'(y_1) \triangleq \frac{\partial \varepsilon_2}{\partial y_1} = \left\{ (y_M - x_M) f(x_M) - \left[\frac{y_M}{4}\, f\!\left(\frac{y_M}{2}\right) + \int_{x_M}^{y_M/2} f(t)dt \right] \left(\frac{\partial y_M}{\partial x_M} \right) \right\}$$

$$\cdot \frac{\partial x_M}{\partial y_{M-1}} \prod_{i=1}^{M-2} \left(\frac{\partial y_{M-i}}{\partial x_{M-i}} \right) \left(\frac{\partial x_{M-i}}{\partial y_{M-i-1}} \right) \qquad (12)$$

for $M > 2$.

We note that most of the calculation of the quantities appearing in the above expressions for the derivative of the error term are a byproduct of the standard calculations performed in the Lloyd-Max algorithm. Thus the update procedure given by (8) does not involve a large amount of additional computations.

IV. NUMERICAL EXAMPLE

For the purpose of illustration, we consider the computation of an optimum quantizer for an input signal having a zero mean and unit variance Gaussian distribution. Letting f(x) denote the corresponding continuous density, we have

$$\frac{\partial x_2}{\partial y_1} = \frac{\sqrt{\pi}\left[1 - \mathrm{erf}\left(-\frac{x_2}{\sqrt{2}}\right)\right]}{\exp\left[-(x_2)^2/2\right] \sqrt{2}\, (x_2 - y_1)},$$

and

$$\frac{\partial x_{i+1}}{\partial y_i} = \frac{(x_i - y_i) e^{-x_i^2/2} + \sqrt{\pi/2} \left(\frac{\partial y_i}{\partial x_i} \right) \left[\mathrm{erf}\left(-\frac{x_i}{\sqrt{2}}\right) - \mathrm{erf}\left(-\frac{x_{i+1}}{\sqrt{2}}\right) \right]}{\left(\frac{\partial y_i}{\partial x_i} \right) (x_{i+1} - y_i)\, e^{-x_{i+1}^2/2}}$$

with

$$\frac{\partial y_i}{\partial x_i} = 2 - \frac{1}{\frac{\partial x_i}{\partial y_{i-1}}}$$

for $2 \le i \le N-1$. If N is even, let $N = 2M$ and define ε_1 via (9) as

$$\varepsilon_1 \triangleq \int_{x_M}^{0} (t-y_M)\frac{1}{\sqrt{2\pi}}\, e^{-t^2/2}\, dt. \tag{13}$$

Thus

$$\varepsilon_1 = \frac{1}{\sqrt{2\pi}}\left(e^{-x_M^2/2}-1\right)-\frac{y_M}{2}\,\mathrm{erf}\left(-\frac{x_M}{\sqrt{2}}\right) ,$$

and

$$\varepsilon_1'(y_1) = \frac{\partial \varepsilon_1}{\partial y_1} = \left\{\left[\frac{(x_M-y_{M-1})}{\sqrt{2\pi}}\, e^{-x_M^2/2} - \mathrm{erf}\left(\frac{-x_M}{\sqrt{2}}\right)\right]\left(\frac{\partial x_M}{\partial y_{M-1}}\right)+\frac{\mathrm{erf}\left(\frac{-x_M}{\sqrt{2}}\right)}{2}\right\} \cdot \prod_{i=1}^{M-2}\left(\frac{\partial y_{M-i}}{\partial x_{M-i}}\right)\left(\frac{\partial x_{M-i}}{\partial y_{M-i-1}}\right)$$

for $M > 2$.

For odd N, let $N = 2M+1$ and define ε_2 via (11) as

$$\varepsilon_2 \triangleq \int_{x_M}^{y_M/2} (t-y_M)\frac{1}{\sqrt{2\pi}}\, e^{-t^2/2}\, dt.$$

We have

$$\varepsilon_2 = \frac{1}{\sqrt{2\pi}}\left(e^{-x_M^2/2}-e^{-y_M^2/8}\right)-\frac{y_M}{2}\left[\mathrm{erf}\left(-\frac{x_M}{\sqrt{2}}\right)-\mathrm{erf}\left(-\frac{y_M}{2\sqrt{2}}\right)\right]$$

and

$$\varepsilon_2'(y_1) = \frac{\partial \varepsilon_2}{\partial y_1} = \left\{\frac{(y_M-x_M)}{\sqrt{2\pi}}\, e^{-x_M^2/2} - \frac{1}{2}\left[\mathrm{erf}\left(-\frac{x_M}{\sqrt{2}}\right)-\mathrm{erf}\left(-\frac{y_M}{2\sqrt{2}}\right)+\frac{y_M}{2\sqrt{2\pi}}\, e^{-y_M^2/8}\right] \cdot \left(\frac{\partial y_M}{\partial x_M}\right)\right\}\left(\frac{\partial x_M}{\partial y_{M-1}}\right)\prod_{i=1}^{M-2}\left(\frac{\partial y_{M-i}}{\partial x_{M-i}}\right)\left(\frac{\partial x_{M-i}}{\partial y_{M-i-1}}\right)$$

for $M > 2$.

Table 1 summarizes the results of the algorithm for N=64. This example took 0.136 seconds of computer time on a CDC cyber 170/750 to obtain the results. The values of ε_1 in (13) and y_1 for each iteration are given in Table 2, and we can see the fairly rapid convergence associated with the algorithm.

We also considered the design of a 1000 level quantizer for an input with a zero mean and unit variance Gaussian distribution and with $\varepsilon_1 = 10^{-7}$ in (13). Our method took 106 iterations.

V. CONCLUSION

A simple method of obtaining a good initial guess and an efficient update procedure for Max's algorithm have been introduced. We find that the s-approximation method presented above is easier and more accurate than the g-approximation method for the Gaussian and Rayleigh densities. We note in passing that we can reduce E_s for small values of N in the s-approximation by using two straight line setments to fit the curve. The Newton-Raphson method used to modify the key parameter does not incur significantly more complicated computation, since most of its calculations are required in Max's scheme.

ACKNOWLEDGEMENT

This research was supported by the Air Force Office of Scientific Research under Grant AFOSR-81-0047.

REFERENCES

1. S.P. Lloyd, "Least squares quantization in PCM," unpublished memorandum, Bell Laboratories, 1957; IEEE Trans. Inform. Th., IT-28, pp. 129-137, March 1982.

2. J. Max, "Quantization for minimum distortion," IRE Trans. Inform. Th., IT-6, pp. 7-12, March 1960.

3. M.D. Paez and T.H. Glisson, "Minimum mean-squared-error quantization in speech PCM and DPCM systems," IEEE Trans. Commun., COM-27, pp. 225-230, April 1972.

4. N.C. Gallagher, "Optimum quantization in digital holography," Appl. Opt., 17, pp. 109-115, January 1, 1978.

5. W.C. Adams and C.E. Giesler, "Quantizing characteristic for signals having Laplacian amplitude probability density function," IEEE Trans. Commun., COM-26, pp. 1295-1297, August 1978.

6. P. Noll and R. Zelinski, "Comments on quantizing characteristic for signals having Laplacian amplitude probability density function," IEEE Trans. Commun., COM-27, pp. 1259-1260, August 1979.

7. J.A. Bucklew and N.C. Gallagher, "A note on the computation of optimal minimum mean-squared error quantizers," IEEE Trans. Commun., COM-30, pp. 298-301, January 1982.

8. P. Fleischer, "Sufficient conditions for achieving minimum distortion in a quantizer," IEEE Int. Conv. Rec., Part I, pp. 104-111, 1964.

9. E.F. Abaya and G.L. Wise, "On the existence of optimal quantizers," IEEE Trans. Inform. Th., IT-28, pp. 937-940, November 1982.

10. K. Hoffman, Analysis in Euclidean Space, p. 395. Prentice-Hall: Englewood Cliffs, NJ, 1975.

x(2) = -3.49236
x(3) = -3.07901
x(4) = -2.79492
x(5) = -2.57187
x(6) = -2.38523
x(7) = -2.22298
x(8) = -2.07829
x(9) = -1.94686
x(10) = -1.82582
x(11) = -1.71312
x(12) = -1.60726
x(13) = -1.50710
x(14) = -1.41175
x(15) = -1.32051
x(16) = -1.23280
x(17) = -1.14814

x(18) = -1.06614
x(19) = -0.96848
x(20) = -0.90884
x(21) = -0.83300
x(22) = -0.75872
x(23) = -0.68581
x(24) = -0.61410
x(25) = -0.54343
x(26) = -0.47365
x(27) = -0.40463
x(28) = -0.33625
x(29) = -0.26839
x(30) = -0.20094
x(31) = -0.13379
x(32) = -0.06685
x(33) = -0.00000

y(1) = -3.74420
y(2) = -3.24052
y(3) = -2.91749
y(4) = -2.67236
y(5) = -2.47139
y(6) = -2.29907
y(7) = -2.14689
y(8) = -2.00969
y(9) = -1.88404
y(10) = -1.76760
y(11) = -1.65864
y(12) = -1.55588
y(13) = -1.45832
y(14) = -1.36518
y(15) = -1.27583
y(16) = -1.18976

y(17) = -1.10652
y(18) = -1.02577
y(19) = -0.94718
y(20) = -0.87050
y(21) = -0.79549
y(22) = -0.72194
y(23) = -0.64968
y(24) = -0.57852
y(25) = -0.50833
y(26) = -0.43897
y(27) = -0.37030
y(28) = -0.30221
y(29) = -0.23458
y(30) = -0.16731
y(31) = -0.10028
y(32) = -0.03341

TABLE 1. Optimal quantizer parameters for Gaussian distribution with 64 output levels.

ITERATION	EPSILON	y_1
1	.00429155	-3.7820873
2	.00111264	-3.7595916
3	.00026347	-3.7492165
4	.00004575	-3.7452219
5	.00000427	-3.7442990
6	.00000026	-3.7442057
7	.00000001	-3.7442000

TABLE 2. Results of the iterations for the design of a 64 level quantizer.

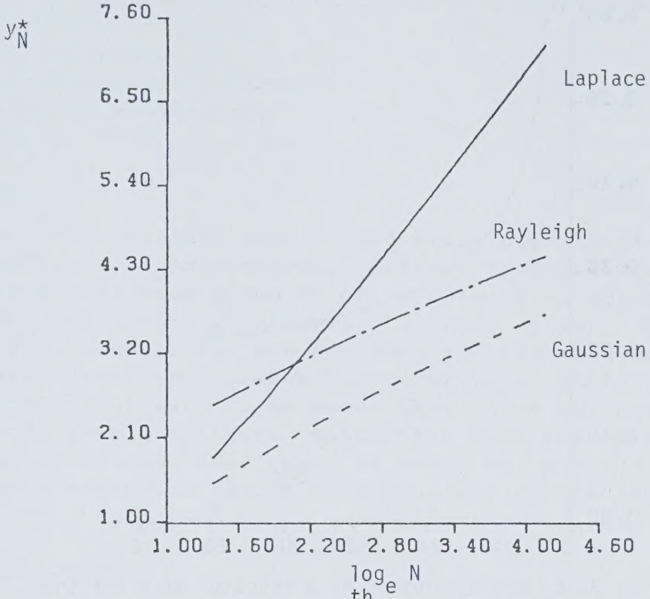


Fig. 1. The tabulated N^{th} output level as a function of the natural logarithm of N, N=4 to 64.

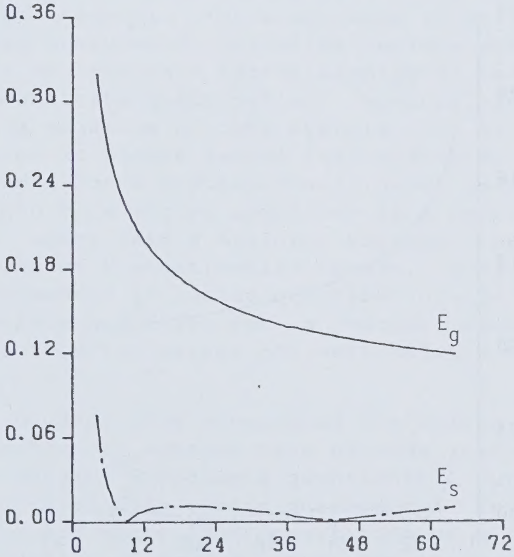


Fig. 2. E_s and E_g plotted as a function of N for the Gaussian density.

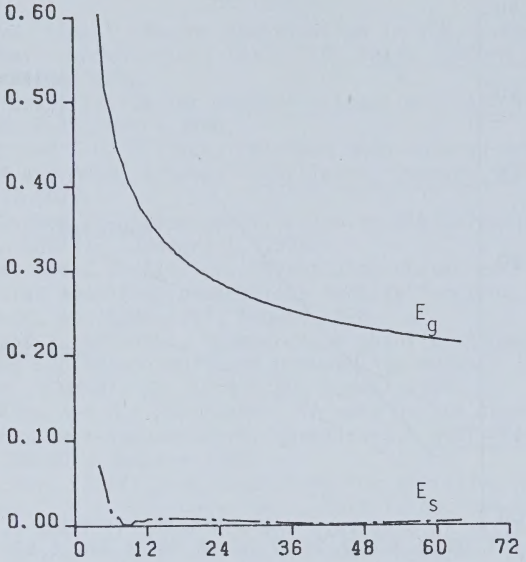


Fig. 3. E_s and E_g plotted as a function of N for the Laplace density.

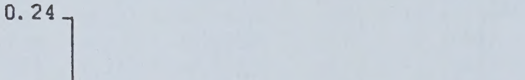


Fig. 4. E_s and E_g plotted as a function of N for the Rayleigh density.

UNIFORM SPHERICAL COORDINATE QUANTIZERS

Peter F. Swaszek
Dept. of Electrical Engineering
University of Washington
Seattle, WA 98102

ABSTRACT

For the minimum mean aquare error quantization of a spherically symmetric source, optimum spherical coordinate quantizers have been shown to often outperform optimum rectangular coordinate quantizers. In this paper, spherical coordinate quantizers are designed and analyzed under the constraint that each scalar quantizer is a uniform (equal step-size) quantizer. It is shown that these uniform spherical coordinate quantizers outperform both the uniform rectangular coordinate quantizers as well as optimum uniform quantizers when the number of dimensions (block length) is small.

I. INTRODUCTION

Quantization is the mapping of a continuously valued random variable onto a finitely valued one. For a scalar input variable, the quantizer partitions the real line into intervals, assigning the same output value to all inputs within the same interval. For a sequence of k inputs, this scalar process produces quantization regions which are the cross products of intervals (parallelepipeds) and is called rectangular coordinate quantization. However, for the quantization of a sequence of data symbols, it is well known that quantizing in blocks rather than symbol by symbol can be advantageous. For a k-dimensional random vector (a data block of length k) a vector quantizer is a partition of the k-dimensional space into N disjoint regions, mapping each region into one of N quantization levels. A major part of the previous research in vector quantizer design has focused upon the optimum quantizer for a vector source, which, unfortunately, is often either not well defined or difficult to implement.

Several studies have considered the design of suboptimum vector quantizers: schemes that provide greater performance than rectangular coordinate quantizers, but with simpler implementation than vector quantizers. In the case of minimum mean square error (MSE) quantizing of a spherically symmetric k-dimensional source, one such implementation was the spherical coordinate quantizer [1-5]. This scheme required that the source symbols be represented in their polar or spherical coordinate form with the resulting coordinates processed by scalar quantizers. The minimization of the MSE included the design of the scalar quantizers for each of the spherical coordinates along with the allocation of levels to each quantizer. The allocation problem deals with the fact that the total number of quantization regions, which is the product of the number of levels for each coordinate, is a specified value N. For rectangular coordinate quantization of identically distributed symbols, symmetry shows that the factorization should be equal across the coordinates. In spherical coordinate quantizing, the coordinates are not identically distributed; hence, the solution to the allocation problem is non-trivial. It was seen that the optimum spherical level allocation was unequal, favoring the longitude most.

In this paper, the usefulness of the spherical coordinate representation of the source will be further demonstrated by considering the quantizer design when the system is constrained to employ uniform scalar quantizers. This restriction has several advantages over optimum spherical coordinates quantization:

- no compressor-expandor pair is needed in order to store or interpret the data. Since no compression occurs, relative values of coordinates are unchanged and arithmetic manipulation of the data is still simple.
- implementation could require only one actual analog-to-digital conversion device The k separate quantizers are just versions of that one device with various scalar multipliers.
- uniform quantizers are known to be insensitive to changes in source statistics. To retain spherical symmetry, these changes must only appear in the magnitude density.

Section II defines the notation and states the assumptions employed throughout this paper. It also contains a more accurrate statement of the problem considered. Section III contains the derivation of the MSE rates for the uniform spherical coordinate quantizers. Section IV provides a comparison of the uniform spherical quantizer performance to uniform rectangular and optimum uniform quantizers. Section V closes the paper with a discussion of results.

II. PRELIMINARIES

In the general k-dimensional case, the spherical coordinates include the magnitude r and k-1 angles ϕ_j (the longitude ϕ_1 and the k-2 latitudes ϕ_j, $j=2,3,\ldots k-1$) [6]. If they are not available as such, the spherical coordinate representation can be computed from the rectangular coordinate representation $\underline{x}=[x_1,x_2,\ldots x_k]^T$ through the following transformations:

$$r = \left[\sum_{i=1}^{k} x_i^2 \right]^{1/2} \quad ; \; r \in [0,\infty)$$

$$\phi_j = \tan^{-1} \left[\frac{x_{j+1}}{\left[\sum_{i=0}^{j} x_i^2 \right]^{1/2}} \right] \quad ; \quad \begin{array}{l} \phi_1 \in [0,2\pi) \\ \phi_j \in [-\pi/2,\pi/2], \; j=2,\ldots k-1 \end{array}$$

The sources under consideration are those that display spherical symmetry. By definition, the k-variate density function in rectangular coordinates for these sources is a function of a quadratic form: $f(\underline{x})=p(\sqrt{\underline{x}^T\underline{x}})$. The source is assumed to have unit variance marginals on the range $[-S,S]$. Applying the transformations above, the density function of the source in the spherical coordinates is seen to be

$$f_k(r) \prod_{j=1}^{k-1} c_j \cos^{j-1}\phi_j \quad ; \quad c_j = \frac{\Gamma[(j+1)/2]}{\Gamma[j/2]\;\Gamma[1/2]}$$

where $f_k(r)$ is the magnitude density

$$f_k(r) = \frac{2\pi^{k/2}}{\Gamma[k/2]} \, p(r)$$

with $p(\cdot)$ the k-variate rectangular density function. We notice that the spherical coordinates are independent random variables. Although this requirement limits the sources of interest, the important independent Gaussian source is included.

Let us assume that the input variable to each scalar quantizer has finite range, $[a,b]$, and is continuously distributed on this range. For an M level uniform quantizer, this range is partitioned into M equal length intervals, each of length $\Delta=(b-a)/M$, and any input which falls into a particular interval is mapped to the midpoint of the interval. For large M, the effects of a uniform quantizer can be approximated by an additive, independent noise source [7] (see Fig. 1). With this model, the scalar quantizer output, $\hat{s}$, is the sum of the input s and a noise random variable e which is independent of the input variable and is uniformly distributed on $[-\Delta/2, \Delta/2]$. The variance of the noise variable can be computed to be $E(e^2)=\Delta^2/12$.

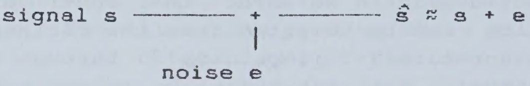


Fig. 1 - Additive noise model.

For the vector input $\underline{x}$, the mean square error (MSE) is defined by

$$MSE = \int_{\mathbb{R}^k} | \underline{x} - \hat{\underline{x}} |^2 \, p(\underline{x}) \, d\underline{x}$$

where hats indicate quantized values. Converting to spherical coordinates and using separate quantizers for each of the spherical coordinates, the MSE becomes:

$$MSE = \int_0^\infty [r^2 + \hat{r}^2 - 2 M_{k-1} r \hat{r}] f_k(r) \, dr \qquad (1)$$

where M_{k-1} is defined sequentially by

$$M_j = \int [\cos \phi_j \cos \hat{\phi}_j + M_{j-1} \sin \phi_j \sin \hat{\phi}_j] \, c_j \cos^{j-1} \phi_j \, d\phi_j \qquad (2)$$

($M_0=1$ and the integral is taken over the range of Φ_j). It can be seen from the above expressions that the MSE for the spherical coordinate quantizers is minimized by maximizing each M_j term over the Φ_j quantizer and by minimizing (1) over the r quantizer (see [5] for further details).

The problem considered in this paper can be stated thus. We desire to minimize (1) under the constraint that the scalar quantizers employed are all uniform quantizers. Uniform quantization requires a finite range for each of the variables; the angle variables already have finite range and we shall consider the radius to have range $[0,R]$. The number of levels N shall be assumed to be large and, since all k quantizers are specified as uniform, the only variables of interest in the design are the numbers of levels alloted to each coordinate. The solution outlined in Section III expresses the MSE as a function of this factorization of N.

III. SPHERICAL MEAN SQUARE ERROR

The MSE for uniform spherical coordinate quantizers is computed below for two cases: optimum allocation of the levels (yielding the smallest MSE) and equal allocation of the levels to each of the spherical coordinates (yielding the simplest implementation).

To begin, we need to solve for the M_{k-1} as a function of the factorization of the number of levels to the phase quantizers. From [1,2,3,5], employing a uniform quantizer for ϕ_1 with N_1 levels yields the value

$$M_1 = \frac{\sin \pi/N_1}{\pi/N_1}$$

For $j>1$, using the additive noise model for a uniform quantizer, (2) becomes

$$M_j \approx \int_{-\Delta_j/2}^{\Delta_j/2} \int_{-\pi/2}^{\pi/2} \left[\cos \phi_j \cos (\phi_j+e) + M_{j-1} \sin \phi_j \sin (\phi_j+e)\right]$$

$$\times \quad \frac{N_j}{\pi} \, c_j \cos^{j-1} \phi_j \, d\phi_j \, de$$

where $\Delta_j=\pi/N_j$ is the step size of the uniform quantizer for the j-th coordinate. Using the trigonometric identities

$$\cos (\phi+e) = \sin \phi \cos e + \sin e \cos \phi$$

$$\sin (\phi+e) = \sin \phi \sin e + \cos e \cos \phi$$

and integrating over e and ϕ_j yields

$$M_j \approx \frac{\sin \pi/2N_j}{\pi/2N_j} \left[\frac{1 + j M_{j-1}}{1 + j} \right] \quad ; \; j>1$$

Since N_j is assumed large, expand the sine into a Taylor expansion, keeping the first two terms only

$$\sin \alpha \approx \alpha - \alpha^3/6$$

so that

$$M_j \approx \left[1 - \frac{\pi^2}{24N_j^2} \right] \left[\frac{1 + j M_{j-1}}{1 + j} \right] \quad ; \; j>1 \qquad (3)$$

Assume that M_{j-1} is of the form

$$M_{j-1} \approx 1 - D_{j-1} \prod_{i=1}^{j-1} N_i^{-2/(j-1)}$$

where D_{j-1} is a function of j only. This holds for $j=2$ with $D_1=\pi^2/6$. Using this result in (3) yields

$$M_J \approx 1 - \frac{\pi^2}{24} N_J^{-2} - \frac{J}{J+1} D_{J-1} \prod_{i=1}^{J-1} N_i^{-2/(J-1)} + \frac{\pi^2}{24} \frac{J}{J+1} D_{J-1} \prod_{i=1}^{J-1} N_i^{-2/(J-1)} N_J^{-2}$$

The terms in this expression are of orders 0, -2, -2, -4 in N_i respectively. Since N was already assumed to be large, we drop the last term to yield

$$M_J \approx 1 - \frac{\pi^2}{24} N_J^{-2} - \frac{J}{J+1} D_{J-1} \bar{N}_J^{-2/(J-1)} N_J^{+2/(J-1)} \tag{4}$$

where $\bar{N}_J$ represents the product of the N_i, $i=1,2,\ldots J$.

Applying the additive noise model to the expression for the MSE in (1) yields

$$MSE \approx \int_{-\Delta_r/2}^{\Delta_r/2} \int_0^{\infty} \left[r^2 + (r+e)^2 - 2 M_{k-1} r (r+e) \right] f_k(r) \frac{N_r}{R} \, dr \, de$$

where $\Delta_r = R/N_r$. Recognizing that the magnitude density has second moment k, since in rectangular coordinates we assumed unit power marginals, the MSE reduces to

$$MSE_{SPH} \approx 2 k (1 - M_{k-1}) + \frac{R^2}{12 N_r^2} \tag{5}$$

OPTIMUM N_J

To minimize the MSE, we optimize (4) and (5) by taking derivatives with respect to the N_i and setting them equal to zero. Optimizing (4) over N_J yields

$$N_{J,OPT} \approx \left[\frac{(J+1)(J-1)\,\pi^2}{24\, J\, D_{J-1}} \right]^{(J-1)/2J} \bar{N}_J^{1/J}$$

and

$$M_{J,OPT} \approx 1 - J \left[\frac{\pi^2}{24} \right]^{1/J} \left[\frac{J\, D_{J-1}}{J^2 - 1} \right]^{(J-1)/J} \bar{N}_J^{-2/J}$$

This result shows that the assumption above was correct with D_J defined sequentially by

$$D_{J,OPT} = J \left[\frac{\pi^2}{24} \right]^{1/J} \left[\frac{J\, D_{J-1}}{J^2 - 1} \right]^{(J-1)/J}$$

Using this result for M_{k-1} in (5) and again, taking derivatives to optimize the MSE, we get

$$N_{r,OPT} \approx \left[\frac{(k-1)\, R^2}{24k\, D_{k-1}} \right]^{(k-1)/2k} N^{1/k}$$

and

$$MSE_{SPH,OPT} = \left[\frac{24kD_{k-1}}{k-1} \right]^{(k-1)/k} \frac{k}{12} R^{2/k} \; N^{-2/k} \qquad (6)$$

EQUAL N_j

In the expressions for M_j in (4) and MSE in (5), we substitute the fact that each N_j equals $N^{1/k}$ to yield

$$M_{j,EQ} = 1 - \left[\frac{\pi^2}{24} + \frac{j\,D_{j-1}}{j+1} \right] N^{-2/j}$$

This result shows that the assumption above was correct with D_j defined sequentially by

$$D_{j,EQ} = \frac{\pi^2}{24} + \frac{j\,D_{j-1}}{j+1}$$

This sequential form can be manipulated to yield

$$D_{j,EQ} = \pi^2 \frac{10 + (j+2)(j+1)}{48(j+1)}$$

Using this result in (5), the MSE simplifies to

$$MSE_{SPH,EQ} = \left[\frac{10+k(k+1)}{24} \pi^2 + \frac{R^2}{12} \right] N^{-2/k} \qquad (7)$$

IV. PERFORMANCE COMPARISON

To demonstrate the advantages of the two forms of uniform spherical coordinate quantizers analyzed above, we compare their MSE rates to those of uniform rectangular and optimum uniform quantizers. Uniform rectangular quantization produces k orthogonal (additive) error terms, each with $MSE=\Delta_i^2/12=S^2/3N_i^2$. By symmetry, N should be factored evenly amongst the rectangular coordinates, yielding

$$MSE_{RECT} \approx \frac{k\Delta^2}{12} = \frac{k}{3} S^2 \; N^{-2/k} \qquad (8)$$

Optimum uniform quantizers are discussed in detail in [8,9] and are defined to be a tesselation of the k-dimensional space by a space-filling polytope P. The MSE for this quantizer is $kG_k(P)$ where $G_k(P)$ is the dimensionless second moment of P. Letting V be the volume of the input space, the MSE can be expressed as

$$MSE_{OPT} = k\, G_k(P) \; V^{2/k} \; N^{-2/k} \qquad (9)$$

with the constant $G_k(P)$ bounded by

$$\frac{\Gamma^{2/k}(1+k/2)}{(k+2)\pi} < G_k < \frac{\Gamma^{2/k}(1+k/2)}{k\pi} \Gamma(1+2/k)$$

The spherical MSE rates are shown in (6) and (7) for optimum and equal level allocation respectively.

We notice that the MSE rates for the quantizers under comparison are dependent upon the domain of definition of the source. The optimum uniform quantizer has rate dependent upon the volume V of the space, not its shape. The spherical quantizers are defined on a hypersphere of radius R with volume $R^k \pi^{k/2}/\Gamma[1+k/2]$. The uniform rectangular quantizers are defined on a hypercube with side of length 2S and volume $(2S)^k$. For comparison, these should all be the same region. To allay these differences, we present results for two cases. Firstly, let the regions all have the same volume; in particular, we consider the hypercube with S=4 (4σ loading) and adjust the radius of the hypersphere. For the second case, we adjust the hypersphere so that it circumscribes the hypercube of side 2S=8. The results from the second case demonstrate the ability of the spherical quantizers since the rectangular region is totally enclosed within it. In both cases, the optimum uniform quantizer's performance is calculated for $V=(2S)^k=8^k$.

Examination of (6)-(9) show that all of the schemes have MSE decreasing at the rate of $N^{-2/k}$; only the coefficients vary. Define the coefficient of MSE as these coefficients divided by k yielding MSE per dimension times $N^{2/k}$. Figures 2 and 3 portray this coefficient of MSE for the two cases described above. We notice that for k<12, the uniform spherical coordinate quantizers compare favorably with the other uniform quantization techniques. For small k, the uniform spherical coordinate quantizers outperform the others.

V. CONCLUSIONS

In this paper, the design of uniform quantizers for the spherical coordinates representation of a spherically symmetric source was considered. Error rates were derived for the large number of levels case. Also, the uniform spherical quantizers were demonstrated to be superior to the uniform rectangular quantizers as well as to the uniform optimum vector quantizers when the number of dimensions was small. The intuitive reason behind the superiority is that the spherical coordinates take advantage of the fact that the source has finite power; hence, having greater probability of occurrence near the origin. The uniform rectangular and optimum uniform quantizers keep the quantization region

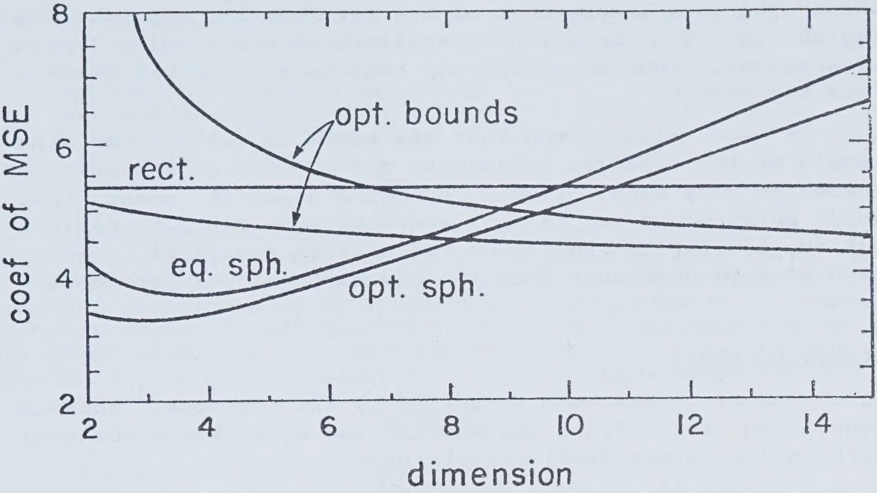


Fig. 2 - MSE rates, equal volume case.

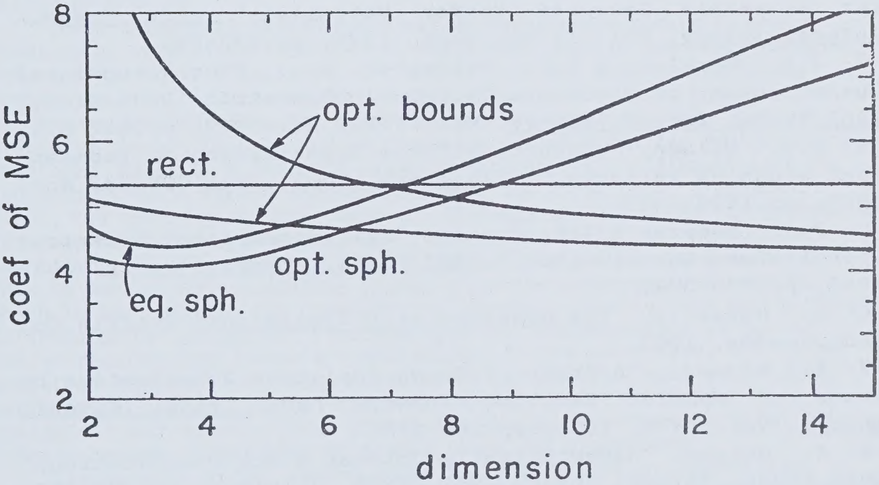


Fig. 3 - MSE rates, circumscribed hypersphere case.

shape the same independent of how far from the origin the region is. The spherical quantizers have diverging angle separations, thus enlarging the regions as they are removed from the origin.

It was also noted that the equal allocation of the levels to the separate quantizers did not reduce the performance by very much. This is exciting since it means that good performance can be achieved with only one A/D device, making the implementation of the uniform spherical quantizers no more difficult than implementing rectangular quantizers.

ACKNOWLEDGEMENT

This research has been supported by the National Science Foundation NEFRI Grant ECS-8206237 and by a Physio-Control Corporation Career Developement Award.

REFERENCES

1- W.A. Pearlman, "Polar Quantization of a Complex Gaussian Random Variable," IEEE Trans. Comm., Vol. COM-27, June 1979 pp.892-899.

2- J.A. Bucklew & N.C. Gallagher Jr., "Quantization Schemes for Bivariate Gaussian Random Variables," IEEE Trans. Inform. Theory, Vol. IT-25, Sept. 1979, pp.537-543.

3- J.A. Bucklew & N.C. Gallagher Jr., "Two-Dimensional Quantization of Bivariate Circularly Symmetric Densities," IEEE Trans. Inform. Theory, Vol.IT-25, Nov.1979, pp.667-671.

4- S.G. Wilson, "Magnitude/Phase Quantization of Independent Gaussian Variates," IEEE Trans. Comm., Vol.COM-28, Nov. 1980, pp.1924-1929.

5- P.F. Swaszek & J.B. Thomas, "Multidimensional Spherical Coordinates Quantization," IEEE Trans. Inform. Theory, July 1983, pp.570-576.

6- M. Kendall, The Geometry of n Dimensions, Griffin Co. Ltd.,London, 1962.

7- B. Widrow, "A Study of Rough Amplitude Quantization by Means of Nyquist Sampling Theory," IRE Trans. Circuit Theory, Vol. CT-3, 1956, pp.266-276.

8- A. Gersho, "Asymptotically Optimal Block Quantization," IEEE Trans. Inform. Theory, Vol.IT-25, July1979, pp.373-380.

9- J.H. Conway & N.J.A. Sloane, "Voronoi Regions of Lattices, Second Moments of Polytopes, and Quantization," IEEE Trans. Inform. Theory, Vol.IT-28, Mar.1982, pp,211-226.

A VECTOR LAPLACIAN QUANTIZER

THOMAS R. FISCHER
Department of Electrical Engineering
Texas A&M University
College Station, TX 77843

ABSTRACT

A vector Laplacian quantizer is designed based on the geometric properties of an independent and identically distributed Laplacian source. It is demonstrated that for large enough dimension, the geometric approach provides a mean absolute-error distortion arbitrarily close to the rate-distortion bound. A vector quantization algorithm is developed based on the integer coordinate vector points on the hyperpyramid. The algebraic structure of the quantizer output points allows a simple implementation.

I. INTRODUCTION

The data compression or vector quantization (VQ) of a continuous-valued source has a rich and varied history, dating to the original work of Shannon [1]. Necessary conditions for the design of optimum scalar quantizers for memoryless sources are well-known [2,3], and the resulting Lloyd-Max quantizer is routinely used in data compression applications. Despite the relative simplicity of the optimum scalar quantizer design, the performance fails to achieve a distortion close to the rate-distortion bound, unless entropy coding of the quantizer output is included in the design process [4].

In an attempt to obtain quantization performance closer to that promised by rate-distortion theory, there has recently been considerable interest in vector quantization. In this case the average statistical properties of a block or vector of data values can be used to advantage. The necessary conditions for an optimum scalar quantizer have been generalized to the vector case and an iterative vector quantizer design algorithm developed based on a training sequence approach [5]. Although such an approach is quite general, it suffers from the three basic drawbacks. First, the training sequence approach is essentially a Monte Carlo method and requires considerable computer time for the design of each vector quantizer. Second, numerous locally optimum designs may be generated. This is particularly sensitive to the algorithm convergence threshold, which in turn effects the computational burden. Finally, the design that is produced has no general structure and the implementation of the VQ design requires considerable computation. In the worst case, the implementation requires a distance calculation between the input vector and every output vector, so that the best representation vector can be assigned. Such a brute force search for the nearest representation point quickly becomes infeasible for modest rates and dimensions.

The purpose of the present paper is to formulate an alternative approach to the design of a vector quantizer for a Laplacian source, based on the geometric properties of the source. The approach is strongly influenced by the clever development by Sakrison [6] of the geometric properties of a Gaussian source. A general formulation of the method is left to another paper; however, the VQ design for an i.i.d. Laplacian source is developed in detail. A source coding theorem is presented for the absolute-error performance measure, demonstrating that for large dimension the encoding method achieves a distortion arbitrarily close to the rate-distortion bound. The proof of the source coding theorem is based on a random coding argument, and hence does not yield a design that is easily implemented. An alternative encoding algorithm is presented with codewords assigned based on a particular geometric structure. This encoding algorithm is quite easy to implement. The resulting performance is evaluated by Monte Carlo simulation and compared to the rate-distortion bound.

II. THE GEOMETRIC STRUCTURE

Let x_i be a sequence of independent and identically distributed (i.i.d.) Laplacian random variables with probability density function

$$p(x_i) = \frac{\lambda}{2} e^{-\lambda|x_i|}. \tag{1}$$

The sequence x_i is assembled into vectors $\underline{x}$ of length L, with a resulting density

$$f(\underline{x}) = \prod_{i=1}^{L} p(x_i).$$

A contour of constant probability density is specified by the condition

$$r \triangleq \sum_{i=1}^{L} |x_i| = ||\underline{x}||_1 = \text{constant}, \tag{2}$$

where $||\underline{x}||_1$ is the familiar ℓ_1 norm. Let S(L,K) be defined as

$$S(L,K) = \{\underline{x}: \sum_{i=1}^{L} |x_i| = K\}. \tag{3}$$

Geometrically, S(L,K) is the surface of a hyperpyramid in L-dimensional space. Scalar random variable r thus indexes a particular contour of density $f(\underline{x})$, or equivalently the pyramid S(L,r).

Random variable r has a probability density function

$$p(r) = \frac{\lambda^L r^{L-1} e^{-\lambda r}}{\Gamma(L)}$$

where $\Gamma(L) = (L-1)!$ Defining

$$\rho = \frac{r}{L}$$

as the per dimension ℓ_1 norm of $\underline{x}$, it is easily verified that

$$E[r] = \frac{L}{\lambda},$$

$$\mathrm{Var}[r] = \frac{L}{\lambda^2},$$

$$E[\rho] = \frac{1}{\lambda},$$

$$\mathrm{Var}[\rho] = \frac{1}{L\lambda^2}.$$

With respect to the (per dimension) ℓ_1 distance measure, it is clear that for large L, vector $\underline{x}$ becomes highly localized around the particular contour of constant density indexed by $r = \frac{L}{\lambda}$. That is, the per dimension ℓ_1 distance between $\underline{x}$ and a point on pyramid $S(L, \frac{L}{\lambda})$ goes to zero as L becomes arbitrarily large. This is derived formally as follows.

Let vector $\underline{x}_p$ be defined as

$$\underline{x}_p = \underline{x} \cdot \frac{E[||\underline{x}||_1]}{||\underline{x}||_1}. \tag{4}$$

Vector $\underline{x}_p$ is not the projection of $\underline{x}$ onto the pyramid $S(L, \frac{L}{\lambda})$, (since $\underline{x}-\underline{x}_p$ is not in general orthogonal to the pyramid surface), but rather a normalized version of $\underline{x}$ so that $||\underline{x}_p||_1 = \frac{L}{\lambda}$. In particular, for all points on $S(L, \frac{L}{\lambda})$ distance $||\underline{x}-\underline{x}_p||_1$ is minimized for $\underline{x}_p$ defined as in (4) (although there may be many points on $S(L,\frac{L}{\lambda})$ that achieve this minimum). The average per dimension absolute error introduced in representing $\underline{x}$ by $\underline{x}_p$ is calculated as

$$E[\frac{1}{L}\,||\underline{x}-\underline{x}_p||_1] = E[|\rho-E(\rho)|] = \frac{2L^L e^{-L}}{\lambda\Gamma(L+1)}. \tag{5}$$

For large L, $\Gamma(L+1)$ may be approximated as

$$\Gamma(L+1) \simeq (L+1)^{L+1}\, e^{-(L+1)} \sqrt{\frac{2}{L+1}}, \tag{6}$$

so that for large L (5) becomes

$$E[\frac{1}{L}\,||\underline{x}-\underline{x}_p||_1] \simeq \frac{2e}{\lambda\sqrt{2\pi(L+1)}}. \tag{7}$$

Thus, for sufficiently large L the per dimension absolute error introduced in representing $\underline{x}$ by $\underline{x}_p$ is negligible. As a result, for large enough L the design of an optimum source code or vector quantizer for the absolute error distortion measure reduces to the problem of optimally allocating codewords (representation points) to the pyramid of parameter $\frac{L}{\lambda}$.

For the mean square error (MSE) distortion measure, the closest point on the pyramid to vector $\underline{x}$ is termed the "projection" of $\underline{x}$ onto the pyramid and is denoted $\hat{\underline{x}}$. For most vectors $\underline{x}$, "projection" $\hat{\underline{x}}$ is the orthogonal projection onto $S(L, \frac{L}{\lambda})$ and it may be calculated in the usual manner. Let vector $\underline{s}$ have components s_i given by

$$s_i = \begin{cases} 1 \text{ if } & x_i \geq 0 \\ -1 \text{ if } & x_i < 0 \, . \end{cases}$$

Then the orthogonal projection of $\underline{x}$ onto the hyperplane containing the pyramid face perpendicular to $\underline{s}$ is given by

$$\hat{\underline{x}}_p = \underline{x} - \left(\tilde{\underline{x}} - \frac{\tilde{\underline{x}}}{\|\tilde{\underline{x}}\|_1} \; \frac{L}{\lambda}\right) = \underline{x} - \underline{s}\left(\frac{\|\underline{x}\|_1}{\|\underline{s}\|_1} - \frac{1}{\lambda}\right) \tag{8}$$

where $\tilde{\underline{x}} = \left(\underline{x}, \frac{\underline{s}}{\|\underline{s}\|_2}\right) \frac{\underline{s}}{\|\underline{s}\|_2}$ and $(\cdot \, , \, \cdot)$ is the usual inner product.

For some vectors $\underline{x}$, the orthogonal projection $\hat{\underline{x}}_p$ obtained by (8) is not on the pyramid; however, $\hat{\underline{x}}_p$ certainly is on the (L-1)-dimensional hyperplane containing the pyramid face orthogonal to $\underline{s}$. Repeatedly applying projection mapping (8) (with different $\underline{s}$) to vector $\hat{\underline{x}}_p$ results in a point that is on the pyramid. This point lies along an edge of the pyramid and is closest to $\underline{x}$ in the Euclidean sense.

The probability that the orthogonal projection, $\hat{\underline{x}}_p$, calculated in (8) does not lie on pyramid $S(L, \frac{L}{\lambda})$ is denoted P_{np}, and may be bounded as

$$P_{np} \leq 2L \int_{\frac{L}{\lambda}}^{\infty} \int_{-(x_L-\frac{L}{\lambda})}^{(x_L-\frac{L}{\lambda})} \cdots \int_{-(x_L-\frac{L}{\lambda})}^{-(x_L-\frac{L}{\lambda})} \frac{\lambda}{2} e^{-\lambda|x_L|} \prod_{i=1}^{L-1} \left(\frac{\lambda}{2} e^{-\lambda|x_i|}\right) dx_1 \ldots dx_L$$

$$= e^{-L} \sum_{r=0}^{L-1} \binom{L}{r+1} \, .$$

Hence P_{np} is bounded by a term which goes to zero as L gets large, so that for large dimension all but a negligible number of vectors $\underline{x}$ have an orthogonal projection (8) that lies on the pyramid.

The per dimension mean square error introduced in approximating $\underline{x}$ by $\hat{\underline{x}}_p$ in (8) may be calculated as

$$E\left\{\frac{1}{L} \|\underline{x}-\hat{\underline{x}}_p\|_2^2\right\} = E\left[\left(\frac{\|\underline{x}\|_1}{L} - \frac{1}{\lambda}\right)^2\right]$$

$$= \text{Var}\,[\rho] = \frac{1}{L\lambda^2} \, .$$

Letting $\hat{\underline{x}}_p$ be the orthogonal projection in (8) and $\hat{\underline{x}}$ the closest point on

$S(L, \frac{L}{\lambda})$ to $\underline{x}$, (so $\underline{\hat{x}} = \underline{\hat{x}}_p$ for most vectors $\underline{x}$) the MSE introduced by projecting $\underline{x}$ onto pyramid $S(L, \frac{L}{\lambda})$ is

$$E\{\frac{1}{L}||\underline{x}-\underline{\hat{x}}||_2^2\} = \underset{\{\underline{x}:\ \underline{\hat{x}}=\underline{\hat{x}}_p\}}{\int \cdots \int} \frac{1}{L}||\underline{x}-\underline{\hat{x}}_p||_2^2 f(\underline{x})d\underline{x} + \underset{\{\underline{x}:\ \underline{\hat{x}}\neq\underline{\hat{x}}_p\}}{\int \cdots \int} \frac{1}{L}||\underline{x}-\underline{\hat{x}}||_2^2 f(\underline{x})d\underline{x}.$$

From the geometry of the orthogonal projection,

$$||\underline{x}-\underline{\hat{x}}||_2^2 \leq 2||\underline{x}-\underline{\hat{x}}_p||_2^2$$

so that the MSE per dimension is bounded by

$$E\{\frac{1}{L}||\underline{x}-\underline{\hat{x}}||_2^2\} \leq 2 \underset{\{\underline{x}:\ \underline{\hat{x}}\ \underline{\hat{x}}_p\}}{\int \cdots \int} \frac{1}{L}||x-\underline{\hat{x}}_p||_2^2 f(x)d\underline{x} + 2 \underset{\{\underline{x}:\ \hat{x}\neq\underline{\hat{x}}_p\}}{\int \cdots \int} \frac{1}{L}||\underline{x}-\underline{\hat{x}}_p||_2^2 \cdot$$

$$f(x)d\underline{x} = \frac{2}{L\lambda^2} .$$

Hence, for large L negligible MSE is introduced in the quantization of $\underline{x}$ by first projecting $\underline{x}$ onto the pyramid of parameter $\frac{L}{\lambda}$, and then quantizing $\underline{\hat{x}}$.

For both mean absolute error (MAE) and mean square error distortion criteria, nothing is lost (for large L) by first representing $\underline{x}$ by an appropriate point on the pyramid, and then quantizing this point with an optimum vector quantizer. Selection of the optimum vector quantizer for points on the pyramid will naturally depend on the particular distortion measure. The essential concept is that only a single contour of constant probability density is important for designing the optimum source code. Further, (somewhat obviously) since the probability density is constant along this important geometric surface, quantizer representation regions should tend to be distributed uniformly on the surface of the pyramid. This intuitive observation is formalized in the following section.

III. A SOURCE CODING THEOREM

The geometric properties of an i.i.d. Laplacian source were developed in section II. This motivated the observation that an optimum source code or vector quantizer (for large L) can be decomposed into the two steps of first finding a point on pyramid $S(L, \frac{L}{\lambda})$ that is close to $\underline{x}$ in the distortion measure sense, and then quantizing this point with an optimum vector quantizer. A source code for points on the pyramid is now derived for the mean absolute error distortion criterion and shown to have a performance that (for large enough L) is arbitrarily close to the rate-distortion bound.

The design of a vector quantizer for points on $S(L, \frac{L}{\lambda})$ depends on the geometric structure of the hyperpyramid. Pyramid S(L, K) encloses an L-dimensional volume, $V(L, K)$, and has an L-dimensional surface area ((L-1)-dimensional volume), $A(L, K)$, that may be calculated to be

$$V(L, K) = \frac{2^L K^L}{\Gamma(L+1)} , .$$

$$A(L, K) = \frac{2^L \sqrt{L}\, K^{L-1}}{\Gamma(L)} . \tag{9}$$

A source code for i.i.d. Laplacian vector $\underline{x}$ is generated as follows.

(i) Form $\underline{x}_p = \underline{x} \cdot \frac{E[||\underline{x}||_1]}{||\underline{x}||_1}$.

(ii) Let a rate, R, be selected and choose 2^{RL} points at random on $S(L, \frac{L}{\lambda})$ according to a uniform distribution. About each point place a representation region of geometric shape $S^*(L, \varepsilon)$ defined by

$$S^*(L, \varepsilon) = \{\underline{x}: ||\underline{x}||_1 \leq \varepsilon\}.$$

Adjust the location of each region (i.e. adjust the centroid) so that the region both contains the original point and has maximum area of intersection with pyramid $S(L, \frac{L}{\lambda})$. Let the centroid of region i be denoted as $\underline{y}_i$.

(iii) If $\underline{x}_p$ falls in the ith region, then $\underline{x}$ is encoded as $\underline{y}_i$.

The absolute error (per dimension) distortion resulting from the encoding of $\underline{x}_p$ using (i)-(iii) may be bounded as

$$E[\frac{1}{L} ||\underline{x}_p-\underline{y}||_1] \leq \frac{\varepsilon}{L} (1-P(\varepsilon)) + \frac{2}{\lambda} P(\varepsilon)$$

where $P(\varepsilon)$ is the probability that $\underline{x}_p$ is not within one of the representation regions. Probability $P(\varepsilon)$ may be calculated as

$$P(\varepsilon) = \prod_{i=1}^{2^{RL}} [1 - \frac{\text{Area of intersection of region i and } S(L, \frac{L}{\lambda})}{\text{Total Area of } S(L, \frac{L}{\lambda})}]$$

or, since the points are selected at random in (ii),

$$P(\varepsilon) = [1 - \frac{\text{Area of intersection}}{\text{Area of } S(L, \frac{L}{\lambda})}]^{2^{RL}} . \tag{10}$$

Since $S^*(L, \varepsilon)$ is the same pyramid shape as $S(L, \frac{L}{\lambda})$, the smallest area of maximum intersection between $S^*(L, \varepsilon)$ and $S(L, \frac{L}{\lambda})$ occurs if a single face of $S(L, \frac{L}{\lambda})$ acts as a hyperplane slicing through the centroid of $S^*(L, \varepsilon)$. The area of intersection is then bounded from below by the (L-1)-dimensional volume of the intersection of $S^*(L, \varepsilon)$ and a (L-1)-dimensional hyperplane parallel to a face of $S^*(L, \varepsilon)$ and passing through the centroid. A useful bound for this volume is

$$\text{Area of intersection} \geq \frac{1}{\sqrt{2}} \; V(L-1, \varepsilon). \tag{11}$$

Using (9) and (11) in (10) yields the bound on $P(\varepsilon)$

$$P(\varepsilon) \leq \left[1 - \frac{\dfrac{2^{L-1}\varepsilon^{L-1}}{\sqrt{2}\,\Gamma(L)}}{\dfrac{2^L\sqrt{L}\,(\frac{L}{\lambda})^{L-1}}{\Gamma(L)}} \right]^{2^{RL}} \tag{12}$$

The rate-distortion function for an i.i.d. Laplacian source is well-known [7] to be

$$R(D) = \log \frac{\frac{1}{\lambda}}{D}$$

so that

$$D = \frac{1}{\lambda} 2^{-R} \tag{13}$$

The distortion in L dimensions, ε, may then be expressed as

$$\varepsilon = \frac{L(1+\delta)}{\lambda} \; 2^{-R}, \; \delta > 0 \; . \tag{14}$$

Substituting (14) into (12) yields

$$P(\varepsilon) \leq \left[1 - \frac{(1+\delta)^{L-1}}{2\sqrt{2L}} \cdot 2^{-R(L-1)} \right]^{2^{RL}} \tag{15}$$

For large L, (15) becomes approximately

$$P(\varepsilon) \simeq \exp\left[- \frac{(1+\delta)^{L-1}\; 2^R}{2\sqrt{2L}} \right].$$

Hence, for arbitrary $\delta > 0$, the probability that $\underline{x}_p$ is not within a mean absolute error distance of ε from some representation point $\underline{y}_i$, goes to zero as dimension L increases. Using (7), it follows that the performance achieveable with the encoding scheme of (i) - (iii) is asymptotically (in L) optimum, and achieves a distortion arbitrarily close to the rate-distortion bound. This derivation is summarized as the following theorem.

Theorem. For the mean absolute-error distortion measure, the source encoding scheme of (i) - (iii) achieves, for large enough dimension L, a distortion arbitrarily close to the rate-distortion bound.

In the case of the mean square error performance criterion, a construction analogous to (i) - (iii) (using the projection, $\hat{\underline{x}}$, instead of $\underline{x}_p$, and selecting the representation regions as spheres instead of pyramids) can be shown to provide asymptotically (in L) optimum performance in the small distortion case. For either performance measure, however, the construction is based on a random assignment of points to $S(L, \frac{L}{\lambda})$, and leads to designs which are extremely difficult to implement. An alternative design approach is to look for points on $S(L,\frac{L}{\lambda})$ that have useful structural properties. This approach is developed in the following section.

IV. INTEGER POINTS ON THE PYRAMID

If pyramid $S(L, \frac{L}{\lambda})$ has $\frac{L}{\lambda} = K$, where K is a positive integer, then it is possible to calculate the number of vectors with integer components that lie on S(L, K). Let N(L, K) be the number of such vectors, defined explicitly as

$$N(L, K) = \text{The number of vectors } \underline{x} \text{, such that}$$

$$\sum_{i=1}^{L} |x_i| = K \text{ and } x_i = \text{integer, for } i=1, \ldots, L. \tag{16}$$

For K = 1, only one of the x_i in (16) is nonzero, with value either 1 or -1. Hence N(L, 1) = 2L. Conversely, if L = 1, then x_i in (16) is either K or -K, so that N(1, K) = 2. Using a combinatorial argument, N(L, K) can be shown to satisfy the recursive formula

$$N(L, K) = N(L-1, K) + N(L-1, K-1) + N(L, K-1) \text{ for } L \geq 2,\ K \geq 2.$$

Since N(1, K) and N(L, 1) are known, it is simple to compute N(L, K).

V. AN ENCODING ALGORITHM

To encode L-dimensional vectors at a specified rate per dimension R, the largest integer value of K should be found such that

$$N(L, K) \leq 2^{RL}. \tag{17}$$

The N(L, K) points of S(L, K) tend to be uniformly spread over the pyramid surface, and make a natural set of representation points for the source code.

The basic encoding algorithm is as follows.

(i) For the MSE quantizer, project $\underline{x}$ onto pyramid $S(L, \frac{L}{\lambda})$ and obtain vector $\hat{\underline{x}}$.

(ii) Scale $\hat{\underline{x}}$ by $\frac{K}{||\hat{\underline{x}}||_1}$ to obtain a corresponding point, $\underline{x}_K$ on $S(L, K)$.

(iii) Find the closest of the $N(L, K)$ integer component vectors on $S(L, K)$ to $\underline{x}_K$. Let the resulting integer component vector be denoted $\hat{\underline{y}}$.

(iv) The vector quantizer representation point is

$$\underline{y} = \frac{\alpha}{K} \hat{\underline{y}}$$

where α is a scale parameter selected to minimize the distortion. Selecting parameter α as

$$\alpha = E[||\hat{\underline{x}}||_1] = \frac{L}{\lambda} \tag{18}$$

provides quantizer output points on pyramid $S(L, \frac{L}{\lambda})$.

VI. PERFORMANCE

The performance of the vector quantization algorithm described in section V was evaluated by Monte Carlo simulation. The average mean square error was calculated for a block of 1000 random (i.i.d. component Laplacian) vectors generated by a random number generator, and then averaged over 100 blocks of vectors with random initial seeds. The results are shown in Table 1. The Monte Carlo simulations verified a performance that was generally expected. The MSE decreased as the dimension increased, but at a very slow rate.

REFERENCES

[1] C. E. Shannon, "A mathematical theory of communication," Bell Syst. Tech. J., vol. 27, pp. 379-423, July 1948.

[2] S.P. Lloyd, "Least squares quantization in PCM," unpublished Bell Laboratories memorandum, 1957; see IEEE Trans. Inform. Th., vol. IT-29, pp. 129-137, March 1982.

[3] J. Max, "Quantizing for minimum distortion," IEEE Trans. Inform. Th., vol. IT-6, pp. 7-12, 1960.

[4] N. Farvardin and J. W. Modestino, "Optimum quantizer performance for a class of non-gaussian memoryless sources," 20th Annual Allerton Conference, Univ. of Illinois, Sept. 1982.

[5] Y. L. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," IEEE Trans. on Comm., vol. COM-28, no. 1, pp. 84-95, Jan. 1980.

[6] D. J. Sakrison, "A geometric treatment of the source encoding of a gaussian random variable," IEEE Trans. Inform. Th., vol. IT-14, no. 3, pp. 481-486, May 1968.

[7] T. Berger, Rate-Distortion Theory, Prentice-Hall, Inc., 1971.

[8] M. D. Paez and T. H. Glisson, "Minimum mean-squared-error quantization in speech PCM and DPCM systems," IEEE Trans. Commun., vol. COM-20, pp. 225-230, Apr. 1972.

[9] P. Noll and R. Zelinski, "Bounds on quantizer performance in the low bit-rate region," IEEE Trans. Commun., vol. COM-26, pp. 300-304, Feb. 1978.

TABLE 1

Mean square error distortion for the pyramid quantizer based on Monte Carlo simulation. Effective rate ($R = \frac{1}{L}\log_2 N(L,K)$) and MSE estimate variance shown in parentheses.

Dimension	MSE (Rate≃1)	MSE (Rate≃2)	MSE (Rate≃3)
1*	0.5	0.1765	0.0514
16	0.308 (R=0.964,Var=2.4×10^{-5})	0.116 (R=1.98,Var=9.5×10^{-6})	0.066 (R=2.97,Var=6.4×10^{-6})
24	0.283 (R=1.002,Var=2.1×10^{-5})	0.101 (R=1.97,Var=2.8×10^{-6})	0.047 (R=2.97,Var=2.5×10^{-6})
32	0.284 (R=1.02,Var=1.0×10^{-5})	0.096 (R=1.97,Var=2.5×10^{-6})	0.040 (R=2.99,Var=1.4×10^{-6})
48	0.276 (R=0.98,Var=8.7×10^{-6})	0.090 (R=1.97,Var=1.0×10^{-6})	0.032 (R=2.99,Var=5.3×10^{-7})
64	0.282 (R=0.96,Var=5.5×10^{-6})	0.085 (R=1.99,Var=1.0×10^{-6})	0.028 (R=2.998,Var=2.3×10^{-7})
100	0.273 (R=0.98,Var=2.6×10^{-6})	0.080 (R=1.99,Var=2.0×10^{-7})	
D(R)**	0.2178 (R=1.)	0.0542 (R=2.)	0.0136 (R=3.)

* Minimum MSE scalar quantizer [8].
**Distortion-rate bound [9].

ON THE PERMUTATION-DECODABILITY OF TRIPLE-ERROR-CORRECTING CODES

S.G.S. SHIVA
Dept. of Electrical Engineering
University of Ottawa
Ottawa, Canada K1N 6N5

A. BENYAMIN-SEEYAR AND V.K. BHARGAVA
Dept. of Electrical Engineering
Concordia University
Montreal, Canada H3G 1M8

ABSTRACT

It is shown that (n,k) binary cyclic triple-error-correcting codes, with n odd and $n \leqslant 2(n-k)+1$, are permutation-decodable. This result is applicable also to codes over $GF(2^m)$.

INTRODUCTION

As is known, permutation-decoding is easy to implement [1,2,3,4]. With reference to an (n,k) binary cyclic t-error-correcting code, let $E(x)$ be an error-polynomial with weight t or less. If there are values of p and q such that $x^p E^{2^q}(x)$ modulo $(1+x^n)$ has degree n-k-1 or less, then in this paper we define $E(x)$ to be permutation-decodable (PD). If this is true for every error $E(x)$, then we define the code to be PD. In this connection, we note that $E_1(x)$ modulo $g(x)$ has weight t or less iff $E_1(x)$ has degree n-k-1 or less, where $E_1(x)$ is any error-polynomial of weight t or less and $g(x)$ is the generator polynomial of the code. This fact provides a way of finding whether or not an error $E(x)$ of weight t or less is PD. Since $(R^2(x)$ modulo $(1+x^n))$ modulo $g(x) = \rho_0^2(x)$ modulo $g(x)$, where the received polynomial $R(x)$ is the sum of a codeword and an error $E(x)$, and $\rho_0(x) = R(x)$ modulo $g(x)$, and since $(x\,R(x)$ modulo $(1+x^n))$ modulo $g(x) = x\rho_0(x)$ modulo $g(x)$, the implementation of permutation-decoding requires essentially a division-circuit needing a shift-register of degree n-k, a squaring-circuit needing a shift-register of degree (2n-2k) modulo n and a threshold gate for determining the weight of the remainder after division. Thus the implementation is easily effected. We also note that, if m is the smallest number such that n divides 2^m-1, then $q = 0,1,2,\ldots, m-1$. If an error $E(x)$ is such that it is PD with $q = 0,1,2,\ldots, L < m-1$, then we say that the error is (L+1)-step PD.

Concerning the available results about the permutation-decodability of codes, it is known [1] that binary (n,k,t) cyclic codes C_t are 1-step PD iff $\frac{k}{n} < \frac{1}{t}$; that is, with just cyclic permutations. This implies that all codes C_1 are 1-step PD. It can also be shown that codes C_2 with $\frac{k}{n} \leqslant \frac{4}{5} - \frac{1}{n}$ are PD. In particular codes C_2 with $\frac{k}{n} < \frac{2}{3}$ are 2-step PD [5]. The derivation of these results involves only the position of the error and, therefore, the results are applicable also to codes over $GF(2^m)$. For example, the (7,3) RS code with $t = 2$ and over $GF(2^3)$ is 1-step PD.

The work reported here was supported in part by the Natural Science and Engineering Research Council of Canada, under grant no. A-3371 and A-4402.

Here, in our analysis for a t-error-correcting (n,k) cyclic code, we represent an n-symbol error pattern E(x) with weight t, by

$$E(x) = \alpha_1+\alpha_2x^{\beta_1}+\alpha_3x^{\beta_2}+\cdots+\alpha_tx^{\beta_{t-1}} \quad , \quad \alpha_i \in GF(2^m). \qquad (1)$$

where $0 < \beta_1 < \beta_2 < \cdots < \beta_{t-1} \leqslant n-1$ are the error positions of n coordinate places of the code. For simplicity in analysis it is assumed one of the errors is fixed at 0 position ($\beta_0 = 0$).

If the code is 1-step PD, then E(x) must satisfy one of the following conditions:

$$\beta_1 \geqslant k+1,$$

or

$$\beta_{i+1}-\beta_i \geqslant k+1 \qquad \text{for any } i = 1,2,\ldots, t-2, \qquad (2)$$

or

$$\beta_{t-1} \leqslant n-k-1.$$

Each of these conditions imply that successive cyclic shifts of the pattern will eventually bring all the errors to the first (or last) n-k positions, and hence the last (first) k positions become error-free symbols; say

$$\underbrace{\overset{0}{x}\ \ x\ \ x\ \text{---}\ x\ \ x\ \ \overset{n-k-1}{x}}_{n-k\text{ positions}}\ \ \underbrace{x\ \ x\ \text{---}\ x\ \ x\ \ \overset{n-1}{x}}_{k\text{ positions}}$$

are the n coordinate places.

If the code is 2-step PD, then E(x), and $E^2(x)$ modulo (x^n+1), patterns according to conditions (2) will be checked, where conditions (2) will also be applicable to $E^2(x)$ modulo (x^n+1) by replacing the β_i error positions with $\beta_i' = 2\beta_i$ modulo n.

In general, if the code is s-step PD, then at most for each successive error patterns of $E(x),(E^2(x),(E^4(x),\ldots,$ and $(E^{2^s}(x))$ all modulo (x^n+1), conditions (2) must be checked, so that with a minimum of squared patterns at least one condition will satisfy, and hence by proper cyclic shifts of the pattern the errors will move out of the last k positions.

It is clear that in the analysis, conditions (2) will be used over and over again.

TRIPLE-ERROR-CORRECTING CODES

Let V be an (n,k) binary cyclic code which is triple-error-correcting and has n odd and $n \leqslant 2(n-k)-1$. Then we first prove the following

Lemma: If n modulo 3 ≠ 0, the code V is PD. If n modulo 3 = 0, the only errors not PD are of the form $x^i(1+x^{n/3}+x^{2n/3})$.

Proof: If the result is true for $n = 2(n-k)-1$, then it is true for also $n < 2(n-k)-1$. Let us, therefore, assume that $n = 2(n-k)-1$. As already stated, codes C_t are 1-step PD if the information rate $<\frac{1}{t}$. Also, if $n-k = \frac{n+1}{2} \leqslant (\alpha,\beta) \leqslant \frac{2n}{3}$, then both 2α and 2β, computed modulo n, are $\leqslant \frac{n}{3}$. In view of these facts, it is sufficient if we consider errors E(x) of the form

$$E(x) = 1 + X^a + X^b,\ 0 \leqslant a \leqslant n-k-1 = \frac{n-1}{2}, \qquad n-k = \frac{n+1}{2} \leqslant b \leqslant \frac{2n}{3}, \qquad (3)$$

where, the error pattern $E(x)$ is cyclically shifted in such a way that all the three errors are embedded between first and $(\frac{2n}{3})$th positions. At this point let us assume, as the first case, that $n-k$ is odd. Then in reference to (3), we have $2a$ modulo $n < n-k$ for $a < \frac{n+3}{4}$ and $2b$ modulo $n \leqslant \frac{n}{3}$. Thus both $2a$ and $2b$ modulo n are $< n-k$ for $a < \frac{n+3}{4}$. This means that we have to investigate further only the numbers

$$\frac{n+3}{4} \leqslant a_1 \leqslant \frac{n-1}{2} \quad , \quad \frac{n+1}{2} \leqslant b_1 \leqslant \frac{2n}{3}. \qquad (4)$$

With $a_2 \triangleq 2a_1$ modulo n and $b_2 \triangleq 2b_1$ modulo n, we have to examine further only the numbers

$$\frac{n+3}{2} \leqslant a_2 \leqslant \frac{2n}{3} \quad , \quad \frac{n+3}{4} \leqslant b_2 \leqslant \frac{n}{3}. \qquad (5)$$

where, we note, a_2 is even and b_2 is odd. In (4), a_1 and b_1 have both odd and even values. This means that $|\{a_2\}| < |\{b_1\}|$ and $|\{b_2\}| < |\{a_1\}|$, where $\{\nu\}$ is the number of values ν can assume. Next, with $a_3 \triangleq 2a_2$ modulo n and $b_3 = 2b_2$ modulo n, we have to consider only the numbers

$$\frac{n+3}{4} \leqslant a_3 \leqslant \frac{n}{3} \quad , \quad \frac{n+3}{2} \leqslant b_3 \leqslant \frac{2n}{3}. \qquad (6)$$

With reference to (6) we note that the difference between any two values of a_3 is a multiple of 4 and that so is the case with b_3. Recalling, with respect to (5), that the difference between any two values of a_2 and the difference between any two values of b_2 are multiples of 2, we conclude that $|\{a_3\}| < |\{b_2\}|$ and $|\{b_3\}| < |\{a_2\}|$. Continuing this line of reasoning, we see that $|\{a_i\}| < |\{b_{i-1}\}|$ and $|\{b_i\}| < |\{a_{i-1}\}|$, where $a_i = 2a_{i-1}$ modulo n and $b_i = 2b_{i-1}$ modulo n. This implies, with reference to (4), that for some values of j, both $2ja_1$ modulo n and $2jb_1$ modulo n are $\leqslant n-k$ or $\geqslant n-k$, unless $a_{i+1} = b_i$ and $b_{i+1} = a_1$. This situation arises when, with reference to (3), n is divisible by 3 and $a = \frac{n}{3}$, $b = \frac{2n}{n}$. This completes the case of $n-k$ being odd. In the case of $n-k$ being even, parallel to (4), we have

$$\frac{n+1}{4} \leqslant a_1 \leqslant \frac{n-1}{2} \quad , \quad \frac{n+1}{2} \leqslant b_1 \leqslant \frac{2n}{3}, \qquad (7)$$

and, parallel to (5), we have

$$\frac{n+1}{2} \leqslant 2a_1 \leqslant \frac{2n}{3} \quad , \quad \frac{n+1}{4} \leqslant 2b_1 \leqslant \frac{n}{3}. \qquad (8)$$

Reasoning further from this point along the same line as in the case of $n-k$ being odd, we arrive at the same conclusion as before. This completes the proof of the Lemma.

By way of illustration, let us consider the (31,15) code which is a subgroup of the (31,16) 3-error-correcting BCH code. We note that $n-k = \frac{n+1}{2} = 16$ which is even. With reference to (7) and (8), we have

$$a_1 = 8,9,10,11,12,13,14,15,$$
$$b_1 = 16,17,18,19,20,$$

and

$$a_2 = 16,18,20, \qquad b_2 = 9,$$

so that

$$a_3 = 9, \qquad b_3 = 18,$$

from which we see that
$4a_3 \bmod 31 = 5$ and $4b_3 \bmod 31 = 10$, both of the values being $< n-k = 16$. This concludes the example.

Now let W be an (n,k) binary cyclic code which is triple-error-correcting and has n odd and $n \leqslant 2(n-k)+1$. Then we prove the following

THEOREM: The code W is PD if n modulo 3 $\neq 0$. If n modulo 3 = 0, the only errors not PD are of the form $x^i(1+x^{n/3}+x^{2n/3})$.

PROOF: Since n is odd, the condition that $n < 2(n-k)+1$ implies that $n \leqslant 2(n-k)-1$. This means that the situation of n being $< 2(n-k)+1$ is covered by the Lemma. Also, the situation of the error being $x^i(1+x^{n/3}+x^{2n/3})$ is covered by the Lemma. Therefore we consider further only the condition $n = 2(n-k)+1$ with n modulo 3 $\neq 0$. Since all t-error-correcting binary cyclic codes, with information rate $<\frac{1}{t}$, are 1-step PD, we have to consider only the errors of the form

$$E_1(x) = 1+x^{\frac{n-1}{2}}, \tag{9}$$

$$E_2(x) = 1+x^u+x^v,\ 0 < u \leqslant \frac{n-3}{2},\ v = \frac{n-1}{2}. \tag{10}$$

With respect to (9) we see that $E_1^2(x)$ modulo $(1+x^n) = 1+x^{n-1}$ which is clearly PD. Concerning (10), we have

$$u_1' = 2u \text{ modulo } n = 2,4,6,\ldots,\ n-3, \qquad v_1' = 2v = n-1. \tag{11}$$

But all errors $1+x^{u_1'}+x^{v_1'}$, with $u_1' > k = \frac{n+1}{2}$, are 1-step PD. Therefore, we have to examine only the numbers.

$$u_1 = 2,4,6,8,\ldots N_1, \qquad v_1 = n-1, \tag{12}$$

where N_1 is a multiple of 2 and $\leqslant \frac{n+1}{2}$.
Using again the argument employed in obaining (12) from (11) we have, from (12),

$$u_2 = 4,8,16,\ldots,N_2,\nu_2, \qquad v_2 = n-2 \tag{13}$$

where u_2 is $2u_1$ modulo n, v_2 is $2v_1$ modulo n, N_2 is the largest multiple of 4 and $\leqslant \frac{n+1}{2}$, $\nu_2 = 0$ if $N_1 < \frac{n+1}{2}$, $\nu_2 = 1$ if $N_1 = \frac{n+1}{2}$. Continuing the process, we have

$$u_3 = 8,16,24,\ldots N_3,\nu_3, \qquad v_3 = n-4 \tag{14}$$

where N_3 is the largest multiple of 8 and $\leqslant \frac{n+1}{2}$ and $\nu_3 = 0$ if $N_2 < \frac{n+1}{2}$, $\nu_3 = 1$ if $N_2 = \frac{n+1}{2}$. From the discussion so far, it is clear that, if we continue the process further, we reach the situation

$$u_{\mu+1} = 2^{\mu+1},\nu, \qquad v_{\mu+1} = n-2^{\mu}, \tag{15}$$

where μ is the largest integer such that $n-2^{\mu} \geqslant k+2 = \frac{n+5}{2}$ and $\nu = 0$ or 1. The error $1+x+X^{n-2^{\mu}}$ is 1-step PD since $n-2^{\mu}-1 \geqslant \frac{n+3}{2} = k+1$. Thus we are left with the numbers

$$u_{\mu+1} = 2^{\mu+1}, \qquad v_{\mu+1} = n-2^{\mu}, \tag{16}$$

for further consideration. With reference to (10) and (16) we see that $(1+x+x^{\frac{n-1}{2}})^{2^{\mu+1}} = 1+x^{2^{\mu+1}}+x^{n-2^{\mu}}$ the computation being modulo $1+x^{n}$. This means that $1+x^{u_{\mu+1}}+x^{v_{\mu+1}}$ is PD. This completes the proof of the Theorem.

Comment: In the case when n modulo 3 = 0, if the permutation-decoding procedure does not result in an error of weight 3 or less, next we find i such that $(R(x)+x^{i}(1+x^{n/3}+x^{2n/3}))$ modulo $g(x) = 0$, where i can be $0,1,2,\ldots\frac{n-3}{3}$. If there is no such i, then we conclude that we have a detectable-only error.

By way of illustration let us consider the (45,23) nonprimitive BCH code which is triple-error-correcting. With reference to (11) we have

$$u_1' = 2,4,6,8,10,\ldots 42, \qquad v_1' = 44,$$

from which

$$u_1 = 2,4,6,8,10,\ldots N_1 = 22, \qquad v_1 = 44,$$

so that

$$u_2 = 4,8,12,N_2 = 16, \qquad v_2 = 43,$$

which, in turn, lead to

$$u_3 = 8,\ N_3 = 16, \qquad v_3 = 41,$$

so that, we finally have, corresponding to (16), that

$$u_4 = 16, \qquad v_4 = 37.$$

We note, with respect to (16), that $\mu = 3$, and that

$$n-2^{\mu} = 45-8 = 37 \geqslant 25 = k+2.$$

In certain applications, codes with an information rate of $\frac{1}{2}$, are required. One such application [6] is in ARQ systems. In this situation a code with minimum distance d is used to correct at most, say, t_1 errors and detect $d-1-t_1$ errors. It would be a great advantage if the error-correcting procedure is simple. In this connection, we note that binary BCH codes with ($n = 2^{m}-1$, $k = \frac{n+1}{2}$) exist for prime values of m, such that the generator polynomial for these codes is

$$g(x) = \mathrm{LCM}(p_1(x) \cdot p_3(x)\ldots p_{d-2}(x))$$

of degree $n-k = \frac{n-1}{2}$. Each of $p_1(x), p_3(x), \ldots, p_{d-2}(x)$ minimal polynomials of α_i, $i = 1,2,\ldots, d-2$, being elements of $GF(2^{m})$, has degree no greater than m, and also divides m [2]. Because m is a prime value, therefore $p_i(x)$ polynomials are all of degree m (all cyclotomic cosets have the same length of m), and m divides $\frac{n-1}{2}$. Hence shortening such a code by 1 bit results in a rate-$\frac{1}{2}$ code. If t_1 is taken to be 3, then this shortened code would be PD according to the Theorem, provided we append a 0 to the codeword so that the cyclic property is restored, before applying the decoding procedure.

PERMUTATION DECODER

A decoder based on the permutation decoding concept can be practically implemented. Consider an (n,k) binary cyclic code with generator polynomial g(x). Suppose that we shift the received polynomial R(x), from higher-order-term-first, into a division circuit with (n-k)-stages shift register to produce $\rho_0(x) = R(x)$ modulo g(x). As soon as the entire R(x) has been shifted into the circuit, the weight of $\rho_0(x)$ in the register is tested by an (n-k)-input threshold gate whose output is "1" when t or fewer of its inputs are nonzero; otherwise, it is "0". This will imply that errors of E(x) are confined to the first n-k check positions, or some errors in E(x) lie in the first n-k and some in the last k positions, respectively. Then the decoding procedures can proceed as follows:

If the code is 1-step PD, then the errors of E(x) will be brought to the first n-k check positions with at most (n-k-1) cyclic shifts of $\rho_0(x)$.

If the code is 2-step PD, and $(x^i \cdot \rho_0(x)$ modulo $(x^n+1))$ modulo g(x) has weight > t, for i = 0,1,..., n-k-1, the threshold gate output is 0. Then $\rho_0(x)$ through a wired squaring circuit is squared and the produced $\rho_0^2(x)$ modulo (x^n+1) polynomial, from higher-order-term-first, is shifted into the division circuit. The weight of $(x^i \cdot \rho_0^2(x)$ modulo $(x^n+1))$ modulo g(x) polynomials are tested. We find the errors with at most 2(n-k-1) cyclic shifts of $x^i \cdot \rho_0(x)$ modulo g(x), and $(x^i \cdot \rho_0^2(x)$ modulo $(x^n+1))$modulo g(x).

In general, if the code is s-step PD, then at most s•(n-k-1) cyclic shifts of $\rho_0(x)$, $\rho_0^2(x)$,..., $\rho_0^{2^s}(x)$ totally will complete the decoding procedures.

As an example, the implementation of a decoder for binary (31,15) BCH code generated by

$$\begin{aligned} g(x) &= p_0(x) \cdot p_1(x) \cdot p_3(x) \cdot p_5(x) \\ &= (1+x)(1+x^2+x^5)(1+x^2+x^3+x^4+x^5)(1+x+x^2+x^4+x^5) \\ &= 1+x+x^4+x^8+x^{11}+x^{12}+x^{15}+x^{16}, \end{aligned}$$

is shown in figure 1.

This permutation decoder, in addition to the binary cyclic codes can be generalized to nonbinary codes in $GF(2^m)$ with some modifications. As an example consider the (31,15) RS code with minimum distance d = 17, over $GF(2^m)$, where m = 5. The field $GF(2^5)$ is formed by a primitive polynomial $p(x) = 1+x^2+x^5$ with α as the primitive element of the field. The generator polynomial g(x) for this code is

$$g(x) = \prod_{i=1}^{16} (x-\alpha^i) = g_{16} \cdot x^{16}+g_{15}x^{15}+\cdots+g_1x+g_0,$$

The feedback multipliers $g_0, g_1, \ldots, g_{16}$ are powers of α. Each nonbinary element of $GF(2^5)$ can be expressed as a 5-tuple over $GF(2)$. Therefore, the hardware implementation of this nonbinary code will be 5-times larger. This implies that to accommodate each symbol we require a 5-stage shift register. For the division circuit part we require addition and multiplication operations, and for the squaring circuit part, squaring operations are needed. These operations must be done on the field elements of $GF(2^5)$ for the selected minimal polynomial $p(x)$, and they can be implemented similarly to those circuits given in References [3 and 7]. The decoding procedures in this case will follow the same arguments as for the binary case.

In addition, each cyclic shift of patterns $GF(2^m)$, will require m-bit shifts at a time.

CONCLUDING REMARKS

We have shown that triple-error-correcting binary (n,k) cyclic codes with n odd and $\frac{k}{n} \leqslant \frac{1}{2} + \frac{1}{n}$ are permutation-decodable if $n \bmod 3 \neq 0$. If n modulo $3 = 0$, the only errors not PD are of the form $x^i(1+x^{n/3}+x^{2n/3})$. Since the derivation of this result depended only on the position of the errors, the result is applicable also to triple-error-correcting cyclic codes over $GF(2^m)$. For example, the (15,7) RS code over $GF(2^4)$ can correct 4 or fewer errors. However, if this code is used to correct 3 or fewer errors, then the only errors not PD are of the form $x^i(\alpha_1+\alpha_2x^5+\alpha_3x^{10})$ where α_1, α_2 and α_3 are elements of $GF(2^4)$. Also, the result is useful in reducing the amount of computation in determining whether or not a triple-error-correcting code, with information rate $> \frac{1}{2} + \frac{1}{n}$, is PD. For example, suppose we use the (51,27) non-primitive BCH code, which is 4-error-correcting, for correcting 3 or fewer errors. Since the information rate is $\frac{27}{51} < \frac{2}{3} < 1$, all single errors are 1-step PD and double-errors are 2-step PD. According to the Theorem, permutation decoding brings all three-errors to the first 25 positions. This means, since $n-k = 24$, we have to consider only the errors of the form $1+X^a+X^{24}$, $0 < a \leqslant 23$. For all even a, the square root of $1+X^a+X^{24}$ corresponds to 3 errors in the first $13 < n-k$ positions. Thus we have to consider further only the numbers $a = 1,3,5,\ldots, 23$; $b = 24$. For $a \geqslant 15$, we have $k < (2a,2b) \leqslant n-1$. Thus we are left with the numbers $a = 1,3,5,\ldots 13$; $b = 24$. For $a = 1,3,5,\ldots,9$; $b = 24$, we have $2b-2a=48-18 = 30 \geqslant k+1=28$. This means a further examination of only the numbers $a = 11,13$; $b = 24$. We have $4a = 44,1$; $4b = 45$, where the computation is modulo n. Here $1+X+X^{45}$ and $1+x^{44}+x^{45}$ are clearly PD. Thus the code is PD except for errors of the form $x^i(1+x^{17}+x^{34})$.

REFERENCES

[1] F.J. MacWilliams, "Permutation Decoding of Systematic Codes", Bell System Technical Journal, Vol. 43, pp. 485-505, 1964.

[2] F.J. MacWilliams and N.J.A. Sloane, "The Theory of Error-Correcting Codes", New York: North Holland, pp. 513-514, 1978.

[3] S. Lin and D.J. Costello, Jr., "Error-Control Coding", Englewood Cliffs, N.J., Prentice-Hall, pp. 125-138, 1983.

[4] G.C. Clark, Jr. and J.B. Cain, "Error Correction Coding for Digital Communications", New York: Plenum Press, pp. 112-119, 1981.

[5] P.W. Yip, S.G.S. Shiva and E.L. Cohen, "Permutation-Decodable Binary Cyclic Codes", Electronics Letters, Vol. 10, pp. 467-468, 1974.

[6] S. Lin and P.S. Yu, "A Hybrid ARQ Scheme with Parity Retransmission for Error Control of Satellite Channels", IEEE Transactions on Communications, Vol. COM-30, pp. 1701-1719, 1982.

[7] E.R. Berlekamp, "Algebraic Coding Theory", New York: McGraw-Hill, pp. 21-69, 1968.

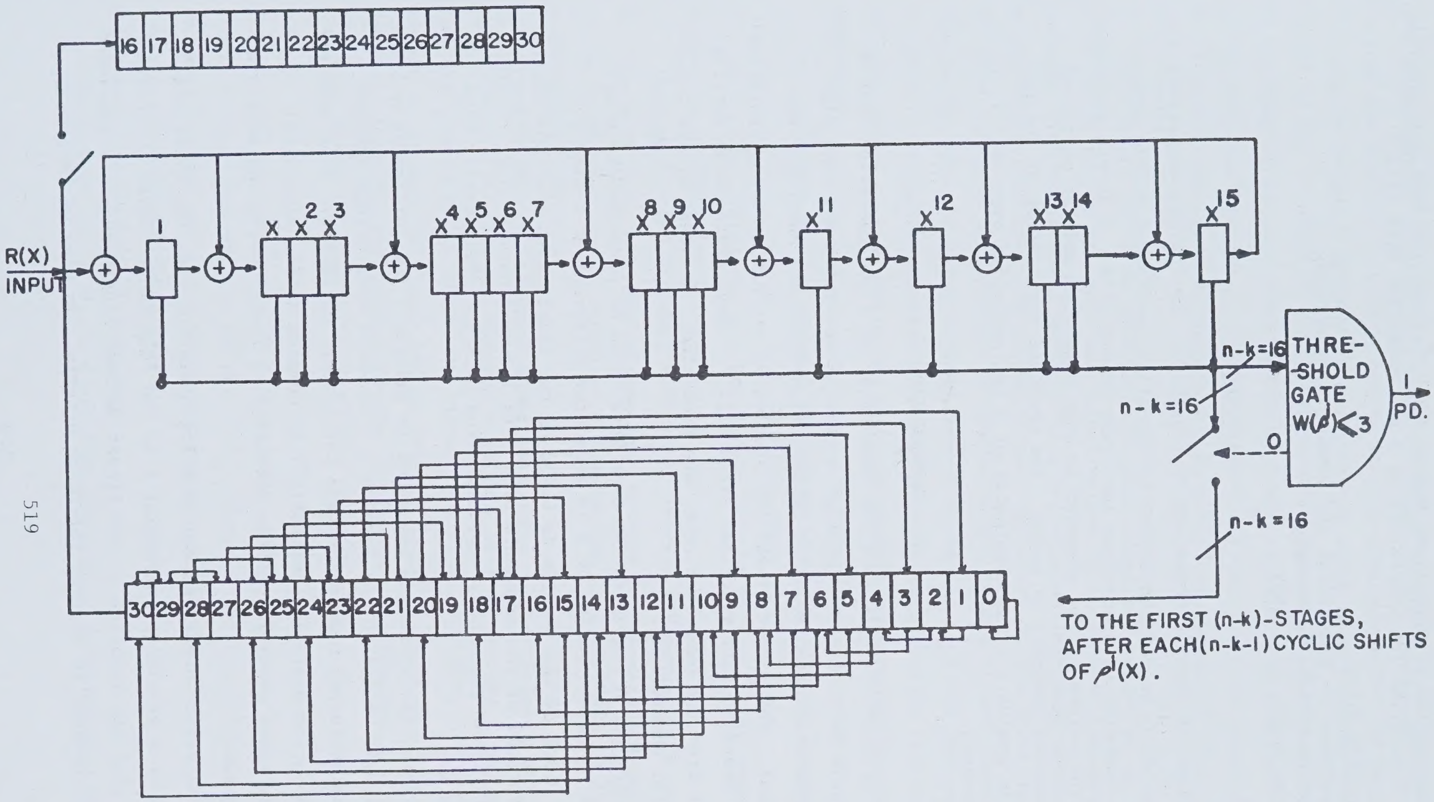

16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
R(X)
INPUT
1
X X^2 X^3
X^4 X^5 X^6 X^7
X^8 X^9 X^10
X^11
X^12
X^13 X^14
X^15
n-k=16
n-k=16
THRE-
-SHOLD
GATE
W(ρ)<3
1
PD.
0
n-k=16
30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
TO THE FIRST (n-k)-STAGES,
AFTER EACH(n-k-1) CYCLIC SHIFTS
OF ρ^1(X) .

FIGURE I. THE IMPLEMENTATION OF THE PERMUTATION DECODER FOR BINARY(31,15) BCH CODE GENERATED BY:

$$g(x) = 1 + X + X^4 + X^8 + X^{11} + X^{12} + X^{15} + X^{16}.$$

PARITY DEFINITION BY OPERATIONS MODULO SOME 2^i-1 AND ITS POSSIBLE APPLICATION IN EVALUATING THE COMPLEXITY OF A LOG OPERATION OVER GF(2^n)

BENJAMIN ARAZI
Department of Electrical and Computer Engineering
Louisiana State University
Baton Rouge, LA 70803

ABSTRACT

A class of numeric sequences of length 2^n-1 is introduced, having a structure which is defined either by operations performed modulo some Mersenne numbers, or operations involving powers of 2. A direct relation between the values of the elements in such sequences and the parity of their location indices, is shown. The possible application of the sequences in evaluating the complexity of a log operation over GF(2^n), is then discussed.

1. INTRODUCTION

The problem of evaluating the complexity of a log operation over a finite field have received recently a special attention and some algorithms have been proposed. The devised algorithms are basically subexponential. This paper treats the problem of performing a log operation over the specific very practical field GF(2^n). Some approaches, totally different from the published ones, are presented. We start by the following trivial observation.

Theorem 1: A log operation over GF(2^n) can be performed by n operations in which the parity of x is found for different $\alpha^x \in$ GF(2^n).

The proof of the theorem is based on the simple fact that the binary representations of the exponents of α^x, $(\alpha^x)^2$, $(\alpha^x)^{2^2}$,..., $(\alpha^x)^{2^{n-1}}$ are cyclic shifts of the binary representation of x and their parity bits then form x.

Utilizing the principle described in Theorem 1 for performing a log operation over GF(2^n) should involve a tool which will bridge between operations performed modulo 2^n-1 and the concept of parity. This also explains why a log operation over GF(2^n) is complex (when using this approach), since every integer is divisible by 2 when operating modulo an odd integer.

One specific interpretation of a log operation over the field GF(2^n) generated by a primitive polynomial f is the following. Given a binary n-tuple, find its location in the linear maximal-length-sequence generated by f, with respect to another reference n-tuple. In view of Theorem 1, the complexity of finding this location index is practically equivalent to finding its parity. A binary linear maximal-length-sequence of periodicity 2^n-1 can be uniquely defined by a process in which binary values are assigned to any location in the sequence where all the locations belonging to the same cyclotomic coset modulo 2^n-1 are assigned equal values. (The cyclotomic coset modulo 2^n-1 of an integer $x \varepsilon [0, 2^n-2]$ consists of all the distinct numbers from the set $\{x, 2x \bmod(2^n-1), 4x \bmod(2^n-1),\ldots, 2^{n-1}x \bmod(2^n-1)\}$.)

This paper first introduces a class of numerical sequences of length 2^n-1, which on the one hand have equal values in locations belonging to the same cyclotomic coset modulo 2^n-1 (like maximal-length-sequences), and on the other hand have the property that the parity of the location indices of their elements is instantly derived from the values of the elements themselves, thus bridging between operations modulo 2^n-1 and the concept of parity. Possible approaches for using these sequences in evaluating an upper bound to the complexity of a log operation over $GF(2^n)$, are then suggested.

2. LOCATION PARITY RECOVERY IN SEQUENCES CONSTRUCTED BY OPERATIONS MODULO 2^n-1

Notation: HW(x) denotes the Hamming weight of the binary representation of a non-negative integer x.

Theorem 2: (a) If $HW(x) \neq HW(x+1)$; $HW(x+2) \neq HW(x+3)$ for a certain x, then x is even. (b) Out of any four successive non-negative integers there is at least one, denoted by x, having the property $HW(x) = HW(x+1)-1 = HW(x+2)-1 = HW(x+3)-2$.

Proof: (a) It is simply shown that $HW(z) = HW(z+1)$ for $z \bmod 4=1$. Since $HW(x) \neq HW(x+1)$ and $HW(x+2) \neq HW(x+3)$ it follows that $x \bmod 4 \neq 1$ and $(x+2) \bmod 4 \neq 1$. Since one integer out of x, x+1, x+2, x+3 modulo 4 equals 1, it must be either x+1 or x+3 and in any case, x is even. (b) It is clear that the specified property of x holds for any $x \bmod 4 = 0$. (It also holds for some other x where $x \bmod 4 = 2$.)

Q.E.D.

Let B_n be a sequence of 2^n-1 elements taken from some field where $(B_n)_i = (B_n)_j$ for $HW(i) = HW(j)$. ($(B_n)_i$ denotes the i-th element of B_n where the first one is no. 0.) For example: the structure of B_5 is abbcbccdbccdcddebccdcddecddedee.

Based on theorem 2(a) we have:

Theorem 3: If a, b, c, d are four successive elements of a sequence

B_n where $a \neq b$; $c \neq d$ then the location index of a (in the sequence B_n) is even.

Since the binary representations of x and $y = 2x \bmod (2^n-1)$, for $0 \leq x \leq 2^n-2$, are obtained one from the other by a cyclic shift, it follows that $HW(x) = HW(y)$ and B_n therefore has equal values in locations belonging to the some cyclotomic coset modulo 2^n-1 and in view of theorem 3, *these sequences have also the property that the parity of the location of some elements in them is instantly derived from the values of the elements themselves.*

It should be noted that for a prime 2^n-1 there are $(2^n-2)/n$ different maximal-length-sequences, and the same number of cyclotomic cosets (excluding the coset $\{0\}$). It can then be simply shown that disregarding their first element, all the binary sequences of length 2^n-1 having equal values in locations belonging to the same cyclotomic coset modulo 2^n-1, including binary B_n, can be expressed as the sum (mod 2) of some maximal-length-sequences. In other words; *the sum of some maximal-length-sequences yield sequences in which the parity of the location indices of the elements is derived by simple means from the values of the elements themselves.*

Some more properties of integers related to the Hamming weight of their binary representation are shown next.

Theorem 4: Any number whose binary representation is of Hamming weight i is obtained from $x = 2^i-1$ by successive operations involving multiplications by various values of the form 2^j modulo various values of the form 2^h-1.

Sketch of the Proof: The multiplications by powers of 2 insert some 0's on the right of the binary representation of the multiplicand. The operations modulo 2^h-1 shift the binary representation cyclically. Successive operations of these forms expand the binary representation of 2^i-1 by inserting 0's into any desired places.

Theorem 5: $HW(x) = HW(x+a \cdot 2^i)$ for any $a \cdot 2^i \leq x < (a+1)2^i$, where a, i are any non-negative integers.

Proof: $x = a \cdot 2^i+y$ where $0 \leq y < 2^i$, and $x+a \cdot 2^i = a \cdot 2^i+y+a \cdot 2^i = (2a) \cdot 2^i+y$. It follows that x and $x+a \cdot 2^i$ have the same bits in the i least significant places in their binary representations. The bits in the rest of the places in the representation of x and $x+a \cdot 2^i$ are the binary representation of a and 2a, respectively.

Q.E.D.

Based on theorems 4 and 5, respectively, the sequences B_n have the following properties.

Property 1. For $i = 1,2,\ldots,n$ the sequence consisting of the first 2^i-1 elements of B_n has equal values in locations belonging to the same cyclotomic coset modulo 2^i-1.

Property 2: Let $B_n^{(i)}$ denote the sequence consisting of the first 2^i elements of B_n, $i = 0,1,\ldots,n-1$. The 2^{i-1} elements of B_n starting with the index 2^i, consist of the last 2^{i-1} elements of $B_n^{(i)}$, and in general: the 2^{i-h} elements of B_n starting with the index $2^i+2^{i-1}+\ldots+2^{i-h+1}$ consist of the last 2^{i-h} elements of $B_n^{(i)}$.

The sequences B_n are the only ones having either property 1 or 2. This is based on the fact that both properties show n random decisions made when constructing a sequence satisfying either property. The number of sequences satisfying either property is therefore the number of all possible sequences B_n. Either property can then form a *definition* of B_n, thus demonstrating the connection between operations performed modulo 2^i-1 and operations involving powers of 2.

A possible way (and probably not the only one) for using the results obtained up to now in evaluating the complexity of a log operation over $GF(2^n)$, is treated next.

3. POSSIBLE USES OF B_N IN EVALUATING THE COMPLEXITY OF A LOG OPERATION OVER $GF(2^n)$

The sequences B_n were defined over any field. From this step onwards we treat only the sequences B_n generated over $GF(2^n)$. Since a sequence B_n can be constructed by assigning random values to the locations with indices 2^i-1, $i=0,1,\ldots,n-1$, with the rest of its elements being then determined uniquely, there are $(2^n)^n = 2^{n^2}$ different sequences B_n over $GF(2^n)$.

Let B_n^* be a sequence B_n having the property $(B_n)_{2^i-1} \neq (B_n)_{2^{i+1}-1}$, $i = 0,1,\ldots,n-2$. (where $(B_n)_j$ denotes the j-th element of B_n). In view of theorems 2b and 3, the parity of z for any given $\alpha^z \varepsilon\ GF(2^n)$ can be recovered by calculating at most the values of $(B_n^*)_{z+t}$, $t = 0,1,\ldots,6$, which then enables performing a log operation (Theorem 1). The complexity of a log operation over $GF(2^n)$ can therefore be determined by the complexity of calculating the value of $(B_n^*)_x$ for a given $\alpha^x \varepsilon\ GF(2^n)$.

Like any function over a finite field, the function $f(\alpha^x) = (B_n^*)_x$ can be presented in a polynomial form for any B_n^*. Let $(B_n^*)_x = \sum_{i=0}^{2^n-2} a_i(\alpha^x)^i$. Adapting an approach for treating this polynomial in evaluating the complexity of a log operation over $GF(2^n)$, is much more promising than treating a polynomial $g(\alpha^x)$ which yields directly the parity of x. This is due to the fact that the structure of g is highly irregular because of the antificalness of the concept 'the parity of the exponent of α' when operating over $GF(2^n)$.

The issue with which we are concerned now is finding some properties which the polynomial's coefficients a_i may possess and which will enable calculating the valve of $\sum_{i=0}^{2^n-2} a_i(\alpha^x)^i$ relatively easily for any $\alpha^x \varepsilon\ GF(2^n)$.

Theorem 6: The value of $\sum_{i=0}^{2^n-2} (B_n)_i(\alpha^x)^i$ is calculated by $O(n^3)$ multiplications over $GF(2^n)$ for any sequence B_n and any $0 \leq x \leq 2^n-2$.

Proof: The calculation of $\sum_{i=0}^{2^n-2} (B_n)_i(\alpha^x)^i$ is performed recursively using property 2, described in the previous section. This calculation is best demonstrated by the following example, which considers the case n=5.

Step 1: Calculate $A = (B_n)_0(\alpha^x)^0$, $B = (B_n)_1(\alpha^x)^1$, $C = (B_n)_2(\alpha^x)^2$, $D = (B_n)_3(\alpha^x)^3$, $E = (B_n)_7(\alpha^x)^7$, $F = (B_n)_{15}(\alpha^x)^{15}$.

Step 2: $G = C+D$, $H = D\cdot(\alpha^x)^3 + E$, $I = E\cdot(\alpha^x)^7 + F$.

Step 3: $J = G\cdot(\alpha^x)^2 + H$, $K = H\cdot(\alpha^x)^6 + I$.

Step 4: $L = J\cdot(\alpha^x)^4 + K$.

Step 5: $F(\alpha^x) = \sum_{i=0}^{30} (B_n)_i(\alpha^x)^i = A+B+G+J+L+(\alpha^x)^8\cdot L + (\alpha^x)^{12}\cdot K + (\alpha^x)^{14}\cdot I + (B_n)_{15}(\alpha^x)^{30}$.

The above demonstration clarifies the fact that for a general n the calculation of F(x) is performed in n steps, each one involving not more than n exponentiations of x, O(n) multiplications and O(n) additions. Since any exponentiation over $GF(2^n)$ is performed by at most 2n multiplications over the field, the theorem is proved.

It should be noted that any B_n is uniquely defined by the n values with location indices 2^i-1, $i=0,1,\ldots,n-1$, which are the only ones needed for performing the calculation of $\sum_{i=0}^{2^n-2} (B_n)_i(\alpha^x)^i$. The fact that the length of B_n grows exponentially with n does not pose therefore any practical difficulty, and the space requirement for calculating $F(\alpha^x)$ is only O(n).

Theorem 6, the arguments preceding it, and theorem 1 lead to the following major result.

Theorem 7: If there exist a sequence B_n, a sequence B_n^* and a primitive polynomial f of degree n such that $\sum_{i=0}^{2^n-2} (B_n)_i \alpha^x = (B_n^*)_x$ for any $\alpha^x \varepsilon$ GF(2^n), where the field is generated by f and where the B_n, B_n^* and f are also known, then a log operation over that field is performed in $O(n^4)$ time and $O(n)$ space.

It can be shown that if the described B_n, B_n^* and f exist, and are also known, then a log operation over a field generated by *any* given primitive polynomial g of degree n is still of complexity $O(n^4)$. (Sketch of the proof of this claim: A log operation over the field generated by g can be performed by first finding a root of g over the field generated by f. This is done with complexity $O(n^3)$ using Rabin's probabilistic algorithm. Performing then two log operations over the field generated by f solves the initial log problem.)

A case which is of a special interest is the one where the sequences B_n are binary. The sequences B_n^* are then the two sequences B_n in which $(B_n)_{2^i-1} = \overline{(B_n)_{2^{i+1}-1}}$, i=0, 1,..., n-2, (where a bar denotes a complement operation). The two sequences B_n^* are one the complement of the other. One of these sequences is actually characterized by the fact that its i-th element (the first is no. 0) consists of the parity of the Hamming weight of the binary representation of i. (This sequence also equals the last column of a Hadamard matrix of dimension $2^n \times 2^n$, with its last element being dropped.)

Definition: a sequence {b} of length m is obtained from a sequence {a} of the same length by *h-decimation* of {a}, if {b} is constructed by taking every h-th element of {a} and repeating the process cyclically until m elements are obtained. h is called *the decimation parameter.*

It should be noted that if (h, m) = 1, the elements of {b} consist of all the elements of {a}. If (h, m) > 1 then {b} consists of repetitions of certain elements of {a}.

Let L_i denote the sequence obtained by i-decimation of a certain linear binary-maximal-length-sequence L of periodicity 2^n-1. (L is then identical to L_1.) For (i, 2^n-1) = 1, the sequence L_i is also a maximal length sequence and $L_i = L_j$ for i and j belonging to the same cyclotomic coset modulo 2^n-1. It can be shown that theorem 7 has the following form when dealing with binary B_n^*.

Theorem 8: Let $\sum_{\{i\}}\oplus L_i = B_n^*$ for some binary linear maximal length sequence L of periodicity 2^n-1 and some B_n^*, where {i} is a set of some decimation parameters. Let $\{I\}$ denote the set of numbers consisting of all the cyclotomic cosets modulo 2^n-1 of which the elements of $\{i\}$ are members. Then a log operation over $GF(2^n)$ is performed in $O(n^4)$ time and $O(n)$ space if there is no x outside $\{I\}$ such that $HW(x) = HW(y)$ for $y \varepsilon \{I\}$.

Showing that theorem 7 takes the form of theorem 8 for a binary B_n^* is based on some manipulations with the Vandermonde matrix (whose ij element is α^{ij}) and the fact that the x-th element of a maximal length sequence generated by a polynomial f is trace (α^x) for α being the root (over $GF(2^n)$) of f.

The existence of the conditions posed in theorem 8 was checked experimentally for n=3,4,5,6,7. *The conditions were found to exist for n=3,4,5.*

Consider that in view of theorem 7 (and its special version in the form of theorem 8), a log operation can be performed by finding a sequence $\{a_i\}$ defined by the relation $\sum_{i=0}^{2^n-2} a_i(\alpha^x)^i = B_n^*$, such that $\{a_i\}$ is also a sequence B_n. In order to have some estimate on the probability that such $\{a_i\}$ exists we choose the more popular and easier to analyze case where 2^n-1 is prime. The number of different cyclotomic cosets and different primitive polynomials is then simply $(2^n-2)/n$ where there are $2^n(2^n-1)^{n-1}$ different sequences B_n^*. The relation $\sum_{i=0}^{2^n-2} a_i(\alpha^x)^i = B_n^*$ defines a sequence $\{a_i\}$ for any B_n^* and any primitive polynomial f (modulo which the operations are performed). Even if we assume that any pair B_n^*; f defines a different sequence $\{a_i\}$, their number is at most $2^n(2^n-1)^{n-1}\cdot(2^n-2)/n$. It can be shown that like the 2^{n^2} sequences B_n, the sequences $\{a_i\}$ also have equal values in locations belonging to the same cyclotomic coset. However, if this common property is the only one by which one has to evaluate the probability that some $\{a_i\}$ equals some B_n, it should be observed that the number of different sequences over $GF(2^n)$ satisfying the described property is $(2^n)^{(2^n-2)/n} = 2^{(2^n-2)}$. Based on the information gathered up to now it can then be concluded that the probability that some $\{a_i\}$ equals some B_n is negligible. It is needed to establish some more specific properties that the $\{a_i\}$ possess, under which they belong to a smaller set of which the B_n (or relatively many of them) are also members. The case of binary B_n^* (in which case the $\{a_i\}$ are also binary) is probably easier to analyze. However, the prospects that some $\{a_i\}$ equals some B_n appear to be gloomy (although this condition was established for n=3,4,5) and the following matters should be looked into when trying to devise other schemes for recovering the x-th element of B_n^*, given $\alpha^x \varepsilon$ GF(2^n).

(a) A search for cases where the sequence $\{a_i\}$ equals a sequence B_n was initiated just by the fact that a polynomial expression whose coefficients form a sequence B_n can be calculated with a low complexity. A vital issue which should be looked into, and which is interesting by itself, is looking for more polynomial expressions over GF(2^n) which can be evaluated with a low complexity.

(b) Instead of aiming at recovering the x-th element of a sequence B_n^*, given $\alpha^x \varepsilon$ GF(2^n), for the purpose of finding later whether or not HW(x) = HW(x+1), one can adopt the following approach. Devise a set $S = \{B_{n1}, B_{n2}, \ldots, B_{nj}\}$ of sequences B_n for $j \leq n$, having the property that for any $0 \leq i \leq n-2$, one of these sequences has different values in the 2^i-1 and $2^{i+1}-1$ locations. (The value of j can be smaller than n since a $B_n \varepsilon S$ can have the described property for several values of i, where for j=1, S contains only a single B_n^*.) Under this extension of the idea of B_n^* one should devise an algorithm where the x-th element of any $B_n \varepsilon S$ should be recovered for any $\alpha^x \varepsilon$ GF(2^n). The fact of whether or not HW(x) = HW(x+1), can then be established by calculating at most the values of some elements in every sequence in S. (It should be remarked that this approach could have an advantage over a direct treatment of B_n^* only for a case where $\sum_{i=0}^{j} B_{ni} \neq B_n^*$.)

Two important general issues which should be looked into are:

(a) An interesting possibility is trying to establish a sub-exponential algorithm for performing a log operation over GF(2^n) which instead of aiming at a direct recovery of x, from a given $\alpha^x \varepsilon$ GF(2^n), like the present algorithms do, should rather aim at recovering the x-th element of some sequence B_n^* (and them recovering x in polynomial time). It is felt that since manipulations with sequences B_n^* involve more 'natural' operations over GF(2^n) (based on their structure which is defined by operations modulo 2^n-1), such a sub-exponential algorithm should be of a lower complexity than the existing ones.

(b) The developments presented up to now deal with a sequence B_n^* and the possibility of recovering its x-th element, given $\alpha^x \varepsilon$ GF(2^n). This is however only *one* possible implementation of the results presented in section 2 where possible general approaches for bridging between operations performed modulo Mersenne numbers and operations using powers of 2, were presented. An open question is whether other possible implementations of these ideas exist other than those described.

FINITE DECIPHERABILITY OF MULTIVALUED ENCODINGS

RENATO M. CAPOCELLI and UGO VACCARO
Istituto di Scienze dell'Informazione, University of Salerno, Salerno, Italy

ABSTRACT

The problem of finite decipherability of multivalued encodings is considered. A necessary and sufficient condition for a multivalued encoding F to admit of a decoding automaton is provided. An algorithm for constructing decoding automata with minimum decoding delay is furnished.

INTRODUCTION

An encoding system is said to be multivalued if there may be two or more codewords corresponding to a single source symbol. Multivalued encodings have been recently considered |1| |2| |3|. They seem to constitute an interesting generalization of ordinary codes. In particular multivalued encodings appear particularly suitable for modeling noisy channels. Indeed the set of codewords corresponding to any source symbol may be considered as the set of codewords in which a single codeword can be changed because of the noise.

Let X a finite nonempty set and let X^+ and X^* be the free semigroup and the free monoid generated by X, respectively. We call the elements of X symbols and the elements of X^+ words. We denote by $l(w)$ the length of the word $w \epsilon X^+$ and by λ the empty word. Given $w \epsilon X^+$ and $p, r \epsilon X^*$, if $pr=w$ then p is a prefix of w and r is a suffix of w and we write $w \geq p$; if $r=\lambda$ $w=p$. For any $Z \subset X^+$ we define

$$B(Z)=\{x \epsilon X^* | \; \exists\, w \epsilon Z \; \exists\, r \epsilon X^* \; [w=xr]\}$$
$$P(Z)=\{x \epsilon X^* | \; \exists\, w \epsilon Z \; \exists\, r \epsilon X^+ \; [w=xr]\}$$
$$T(Z)=\{x \epsilon X^+ | \; \exists\, w \epsilon Z \; \exists\, p \epsilon X^+ \; [w=px]\} .$$

Let $x \epsilon X$ and $Z \subset X^*$. With $Z \cdot x$ we denote the set $Z \cdot x=\{y \epsilon X^+ | \; y=zx \text{ and } z \epsilon Z\}$. With $x^{-1} \cdot Z$ we denote the set $x^{-1} \cdot Z=\{y \epsilon X^* | \; xy \epsilon Z\}$.

Let C be a finite subset of X^+ and let R be an equivalence relation on C. An encoding system F is given by C (the set of codewords) and by an equivalence relation on it: $F \equiv (C,R)$. If an equivalence class exists which consists of more than one element, F is a multivalued code; otherwise F=C is an ordinary (single valued) code. In other words, given a finite set A of source symbols, a multivalued code is any mapping F from A into 2^{X^+}. We expand the domain of F from A to A^* in the following manner

1) $F(\lambda)=\{\lambda\}$
2) $\forall x \epsilon A^*, \; \forall y \epsilon A \quad F(x \cdot y)=F(x) \cdot F(y)=\{\alpha\beta | \; \alpha \epsilon F(x) \text{ and } \beta \epsilon F(y)\}$.

Since, in the case of multivalued encodings, unique decipherability is not equivalent to unique decomposability, some new ideas are needed to solve problems related to multivalued encodings. Sato |1| studied the problem of unique decipherability and provided an algorithm for determining whether a given multivalued encoding is uniquely decipherable. A multivalued encoding is uniquely decipherable if for any $x,y \epsilon A^*$ $(x \neq y) \Rightarrow F(x) \cap F(y)=\emptyset$, that is, if no two strings x and y of source symbols have a common encoding. Sato proposed also a conjecture for determining whether a multivalued encoding has the finite delay property. A uniquely decipherable multivalued encoding F is decipherable with finite delay P iff P is the smallest integer

---

Some of the results contained in this paper was presented at IEEE Symposium on Information Theory, 1983.

for which the nondeterministic generalized sequential machine (gsm) M that implements F has an inverse machine M^{-1} such that the serial connection $M^{-1}M$ of M in its initial state and M^{-1} in its initial state amounts to a delay machine with (maximum) delay P. A nondeterministic sequential machine D is called a delay machine with (maximum) delay P iff for some given P and for any arbitrary input $x\varepsilon A^{+}$, $l(x)>P$, any associate output y is a prefix of x, with $l(x)-l(y)\leq P$. In other words, F is decipherable with delay P iff the individuation in a message of P+n initial consecutive codewords suffices to determine the first n symbols of the source sequence that generated it. Notice that, although conceptually different, the above definition for ordinary codes reduce to the standard definition of finite decipherability. Capocelli |2| considered the problem of finite decipherability of multivalued encodings and proved Sato's conjecture to be true. The three properties of unique decipherability, decipherability with finite delay, and synchronizability are central to coding theory and have been extensively described and studied (see |2| and references therein quoted). As is well known they are related to the possibility of decoding a finite message, an infinite message with known starting point, and an infinite message without knowing its beginning and end.

In this paper the problem of decoding multivalued encodings is considered. A necessary and sufficient condition for the existence of decoding automata for multivalued encodings is provided and an algorithm for constructing decoding automata for multivalued encodings is proposed. For single valued codes such decoding automata reduce essentially to those usually considered for decoding ordinary codes (see for example |4|).

FINITE DECIPHERABILITY OF MULTIVALUED ENCODINGS

Following Sato |1|, we can represent a multivalued encoding as a transduction by a finite-state nondeterministic machine and formalize F=(C,R) as a relation between input sequences and output sequences by using a parameter set P^{+}

$$F\equiv\{(\sigma(p),\tau(p))\mid p\varepsilon P^{+}\}$$

$$\sigma(P)=A,\ \tau(P)=F(A)=C,\ \sigma(P^{+})=A^{+},\ \tau(P^{+})=F(A^{+})=C^{+},$$

where $\sigma:T^{+}\longrightarrow A^{*}$ and $\tau:T^{+}\longrightarrow X^{+}$ are mappings onto the input (a shortening homomorphism) and onto the output (a length-preserving homomorphism), respectively, T is the set of the transitions in the state diagram of the finite machine that realizes F, P is the set of all elementary succesfull transition sequences (i.e. of all transition sequences that produce a codeword), and P^{+} is the set of all possible transition sequences. It is also useful to introduce by the relations

$$g\equiv\{(\sigma(p),p)\mid p\varepsilon P^{+}\},\quad h\equiv\{(\tau(p),p)\mid p\varepsilon P^{+}\}\ ,$$

the inverse mappings $\sigma_{P}^{-1}:A\longrightarrow 2^{P^{+}}$ of σ_{P}, and the inverse mapping $\tau_{P}^{-1}:C^{+}\longrightarrow 2^{P^{+}}$ of τ_{P}, respectively, where σ_{P} and τ_{P} are the same as σ and τ except that their domain is limited to P^{+}. Clearly $F=\tau g$, $C=F(A)=\bigcup_{x\varepsilon A}F(x)$, $\bigcup_{x\varepsilon A}g(x)=\bigcup_{x\varepsilon C}h(x)=P$, and $\bigcup_{x\varepsilon A^{+}}g(x)=(\bigcup_{x\varepsilon A}g(x))^{+}=\bigcup_{x\varepsilon C^{+}}h(x)=P^{+}$.

Example 1

Let A={0,1}, X={a,b}, C={aa,ab,aab,bab}. Let us consider the multivalued encoding F(0)={aab,aa}, F(1)={ab,bab} and denote by T the set $T=\{t_1,t_2,t_3,t_4,t_5,t_6,t_7,t_8,t_9,t_{10}\}$. It then results that $g(0)=\{t_1t_2t_3,t_4t_5\}$, $g(1)=\{t_6t_7,t_8t_9t_{10}\}$, $\sigma(t_1)=\sigma(t_4)=0$, $\sigma(t_6)=\sigma(t_8)=1$, $\sigma(t_i)=\lambda$ for $i\neq1,4,6,8$; $\tau(t_1)=\tau(t_2)=\tau(t_4)=\tau(t_5)=\tau(t_6)=\tau(t_9)=a$, $\tau(t_3)=\tau(t_7)=\tau(t_8)=\tau(t_{10})=b$, $h(aab)=t_1t_2t_3$, $h(aa)=t_4t_5$, $h(ab)=t_6t_7$, $h(bab)=t_8t_9t_{10}$. Pratically, in the machine that realizes F every transition arrow is segmented with every output character by adding temporary states. ■

If we denote by D the mapping $\sigma h: C^{+} \longrightarrow 2^{A^{+}}$, a multivalued encoding is uniquely decipherable iff $DF(x)=\{x\}$ for any $x \varepsilon A^{+}$. On the other hand, if we define the sets

$$R \equiv \{x \varepsilon (T \times T)^{+} |\ \alpha_1(x) \varepsilon P^{+} \wedge \alpha_2(x) \varepsilon P^{+} \wedge \tau\alpha_1(x)=\tau\alpha_2(x)\}$$
$$S \equiv \{x \varepsilon (T \times T)^{+} |\ \sigma\alpha_1(x)=\sigma\alpha_2(x)\}\ ,$$

where α_1 and α_2 are the two (projection) homomorphisms defined by $\alpha_1(t_i,t_j)=t_i$, $\alpha_2(t_i,t_j)=t_j$, $(t_i,t_j) \varepsilon (T \times T)^{+}$, a multivalued encoding is uniquely decipherable iff $R \subseteq S$ |1|. R is a regular set, S is a context-sensitive set, and it is decidable whether an arbitrary multivalued encoding is uniquely decipherable. For any $x \varepsilon (T \times T)^{+} \equiv \Gamma^{+}$ let us define the remainder of x (by homomorphisms $\sigma_1=\sigma\alpha_1$ and $\sigma_2=\sigma\alpha_2$) $\mathrm{Rem}: x \varepsilon (T \times T)^{+} \longrightarrow \mathrm{Rem}(x) \varepsilon A^{+} \times \{\lambda\} \cup \{\lambda\} \times A^{+} \cup \{(\lambda,\lambda)\} \cup \{\emptyset\}$ as follows

$$\mathrm{Rem}(x)=\begin{cases} (y,\lambda), & \text{when } \sigma_1(x)=\sigma_2(x)y, \\ (\lambda,y), & \text{when } \sigma_1(x)y=\sigma_2(x), \\ (\lambda,\lambda), & \text{when } \sigma_1(x)=\sigma_2(x), \\ \emptyset & \text{otherwise.} \end{cases}$$

It easy to show the following properties of Rem(x) |1|

$$\mathrm{Rem}(x)=\mathrm{Rem}(y) \Leftrightarrow (\forall z \varepsilon \Gamma^{+})(\mathrm{Rem}(xz)=\mathrm{Rem}(yz))$$
$$\mathrm{Rem}(x)=(\lambda,\lambda) \Leftrightarrow (\forall z \varepsilon \Gamma^{+})(\mathrm{Rem}(xz)=\mathrm{Rem}(z))$$
$$\mathrm{Rem}(x)=\emptyset \Leftrightarrow (\forall z \varepsilon \Gamma^{+})(\mathrm{Rem}(x)=\emptyset).$$

Further, let us consider the minimal-state finite automaton $A_R=\{K_R \cup \{Q_\varphi\}, \Gamma, \delta_R, Q_o, F_R\}$ that accepts the regular set R, where K_R is the internal state set apart from Q_φ, Q_φ is a $\emptyset$ state (a state not in F_R such that for any $x \varepsilon \Gamma$ $\delta_R(Q_\emptyset, x)=Q_\emptyset$), and $\{Q_\emptyset\}=\emptyset$ if such a state does not exist, and where Γ, δ_R, Q_o, and F_R are the input alphabet, the transition function, the initial state, and the final state set, respectively. Finally, let us denote by $A(Q_i)$ the set of all input strings that bring A_R from its initial state Q_o to the state Q_i and by $A_k(Q_i)$ the set of strings of length less than or equal to k that bring A_R from Q_o to Q_i. The following results hold:

Theorem 1 |1|. A necessary and sufficient condition for a finite multivalued encoding to be uniquely decipherable is that the remainders of the elements of $\bigcup_{Q_i \varepsilon F_R} A(Q_i)$ are all (λ,λ). ■

Lemma 1 |1|. Let n be the cardinality of K_R. If for all $Q_i \varepsilon K_R$, the remainders of all the elements of $A_n(Q_i)$ take the same value, then the remainders of all the elements of $A(Q_i)$ take this same value (denoted by $Rem(Q_i)$).

■

Theorem 2 |1|. A necessary and sufficient condition for a finite multivalued encoding to be uniquely decipherable is that for all $Q_i \varepsilon K_R$ the remainders of all the elements of $A_n(Q_i)$ take the same value and that also the remainders of the elements of $\bigcup_{Q_i \varepsilon F_R} A_n(Q_i)$ are all (λ).

■

Let us now consider the minimal-state finite automaton $A_P=\{K_P \cup \{q_\emptyset\}, T, \delta_P, q_o, F_P\}$, that accepts the regular set P^+, $q_\emptyset$ is a $\emptyset$ state, δ_P is the transition rule, and the final state set F_P in fact consists of only the initial state q_o. Let us construct the incompletely defined automaton A $\{K_P \times K_P, T \times T, \delta, (q_o,q_o), F_P \times F_P\}$ where

$$\delta \text{ is } \begin{cases} \delta((q_i,q_j),(t_h,t_k))=(\delta_P(q_i,t_h), \delta_P(q_j,t_k)) \text{ when} \\ \delta_P(q_i,t_h)\neq q_\emptyset, \delta_P(q_j,t_k)\neq q_\emptyset \text{ and } \tau(t_h)=\tau(t_k); \\ \\ \text{not defined otherwise.} \end{cases}$$

Here $(q_i,q_j)\varepsilon K_P \times K_P$, $(t_h,t_k)\varepsilon T \times T=\Gamma$. It is easily seen that A accepts the set R above defined. However, A is not generally a minimal-state machine, and usually many states exists that cannot be reached from the initial state or from which the final state cannot be reached. Let m the number of states of A that can be reached from the initial state $(q_o,q_o)\equiv Q_o$. Let $A_k(Q_i)$ be the set of strings with length less than or equal to k that bring A from Q_o to the state $(q_h,q_k)\equiv Q_i$. For each state Q_i let us define $Rem(Q_i)$ as follows: when the remainders of all the elements of $A_m(Q_i)$ take the same value, $Rem(Q_i)$ take this value; otherwise $Rem(Q_i)=\emptyset$.

We remark that for any state from which the final state can be reached, the remainder can be uniquely assigned and, if the multivalued encoding is uniquely decipherable, is different from $\emptyset$, whereas in general it cannot be uniquely assigned for states from which the final state cannot be reached |1|. These states play an important role in the determination of the finite delay property.

Def. 1 . A loop of states has remainder $\emptyset$ if $x,y\varepsilon\Gamma$ exist such that $\delta(Q_o,x)=Q_i$, $\delta(Q_i,y)=Q_i$, $x,xy\varepsilon A_m(Q_i)$, with either $Rem(x)\neq Rem(xy)$ or $Rem(x)=Rem(xy)=\emptyset$.

The following results hold

Lemma 2 |3|. If for all states Q_i that can be reached from Q_o and $\forall\alpha,\alpha\beta\varepsilon A_m(Q_i)$ it results

$\mathrm{Rem}(\alpha)=\mathrm{Rem}(\alpha\beta)\neq\emptyset$ then $\forall x\varepsilon A_m(Q_i)$ $\forall xy\varepsilon A(Q_i)$ it holds $\mathrm{Rem}(x)=\mathrm{Rem}(xy)\neq\emptyset$. ■

Theorem 3 |2||3|. A necessary and sufficient condition for a uniquely decipherable multivalued encoding F to have the finite delay property is that no loop of states exists with remainder $\emptyset$ in the state diagram of A. ■

For any $x\varepsilon A_m(Q_i)$ consider $\sigma_1(x)$ and $\sigma_2(x)$. Denote by L the length of the longest of its sequences $\sigma_1(x)$ and $\sigma_2(x)$ and by l the length of the longest common prefix of $\sigma_1(x)$ and $\sigma_2(x)$.

Corollary 1 |2||3|. The delay of a finite multivalued encoding F decipherable with finite delay is $P=\max\mu\leq\max L$, where $\mu=L-l$. The maximization is over all $Q_i\varepsilon K_P\times K_P$ of A. ■

DECODING AUTOMATA FOR MULTIVALUED ENCODINGS

Given a multivalued code $F=(C,R)$ consider $\beta\varepsilon B(C^+)$, define an initial decomposition of β a sequence $\delta(\beta)=w_1w_2\ldots w_h\overline{\beta}$, $(h\geq 0)$, such that $\beta=\delta(\beta)$, $w_i\varepsilon C$, $\overline{\beta}\varepsilon P(C)$. For any initial decomposition of $\beta\varepsilon B(C^+)$, say $\delta_i(\beta)=w_{i_1}w_{i_2}\ldots w_{i_{h_i}}\overline{\beta}_i$, consider the corresponding source sequence $D_i(\beta)=a_{i_1}a_{i_2}\ldots a_{i_{h_i}}x$ with $a_{i_j}\varepsilon A$, $w_{i_1}w_{i_2}\ldots w_{i_{h_i}}\varepsilon F(a_{i_1}a_{i_2}\ldots a_{i_{h_i}})$ and $x\varepsilon A\cup\{\lambda\}$ ($x\varepsilon A$ if all codewords beginning with $\overline{\beta}_i$ belong to $F(x)$, $x=\lambda$ otherwise); call $D_i(\beta)$ an initial deciphering of β. Then for any β define a mapping $G:B(C^+)\longrightarrow A^*$ and a mapping $H:B(C^+)\longrightarrow 2^{B(C^+)}\times 2^{T(C)}$ as follows:

$G(\beta)=a_1a_2\ldots a_s\,\varepsilon\, A^*$, where $a_1a_2\ldots a_s$ $(s\geq 0)$ is the longest common prefix of all initial deciphering of β;

$H(\beta)=(R_1(\beta),R_2(\beta))$, where $R_1(\beta)=\{x\varepsilon B(C^+)\mid\exists\, w_1w_2\ldots w_s\varepsilon F(G(\beta)),\exists\,\delta_i(\beta)\ [\delta_i(\beta)=w_1w_2\ldots w_sx]\}$ and $R_2(\beta)=\{x\varepsilon T(C)\mid\exists\, w_1w_2\ldots w_s\varepsilon F(G(\beta)),\ \exists\,\delta_i(\beta)\ [\delta_i(\beta)x=w_1w_2\ldots w_s]\}$.

We may call $R_1(\beta)$ the set of dangling suffixes of β (with respect to G), $R_2(\beta)$ the set of suffixes borrowed by β (with respect to G). Further let us extend the domain of G and H. Let $Y\subset B(C^+)$, $Z\subset T(C)\cup\{\lambda\}$ and let Y and Z not be both empty, define

$$G(Y,Z)=\begin{cases}\text{the longest common prefix, if any, of all sequences } G(\beta_i),\ \beta_i\varepsilon Y, \text{ if } Z=\emptyset;\\ \\ \lambda \quad \text{if } Z\neq\emptyset,\end{cases}$$

and

$$H(Y,Z)=\begin{cases}(R_1(Y),R_2(Y)) & \text{if } Z=\emptyset\\ (Y\cup\{\lambda\}\ ,\ Z-\{\lambda\}) & \text{if } \lambda\varepsilon Z\\ (Y,Z) & \text{in the other cases}\end{cases}$$

where

$R_1(Y)=\{x\varepsilon B(C^+)\mid \exists\, w_1w_2\ldots w_k\varepsilon F(G(Y,\emptyset)), \exists\, \beta_j\varepsilon Y,\ \exists\, \delta_i(\beta_j)\ [\delta_i(\beta_j)=w_1w_2\ldots w_kx]\}$,

$R_2(Y)=\{x\varepsilon T(C)\mid \exists\, w_1w_2\ldots w_k\varepsilon F(G(Y,\emptyset)), \exists\, \beta_j\varepsilon Y,\ \exists\, \delta_i(\beta_j)\ [\delta_i(\beta_j)x=w_1w_2\ldots w_k]\}$.

Finally for $x\varepsilon X$ such that $(Y\cdot x\cap B(C^+),\ x^{-1}Z)\varepsilon 2^{B(C^+)}\times 2^{T(C)\cup\{\lambda\}}-(\emptyset,\emptyset)$ define $G(Y,Z,x)= G(Y\cdot x\cap B(C^+),\ x^{-1}Z)$, $H(Y,Z,x)=H(Y\cdot x\cap B(C^+),\ x^{-1}Z)$. G and H satisfy the following

Lemma 3 |3| . $\forall b\varepsilon X,\ \ \beta\varepsilon B(C^+),\ \beta b\varepsilon B(C^+)$

$$G(\beta b)=G(\beta)\cdot G(H(\beta),b) \tag{1}$$

$$H(\beta b)=H(H(\beta),b) \tag{2}$$

■

Let $D=(S,s_o,X,A,f,\phi)$ be a deterministic, finite, partially defined automaton where S is the set of states, s_o is the initial state, X and A are the input and the output alphabet respectively, f is the function of transition and ϕ is the output function. Given a multivalued encoding $F:A\longrightarrow 2^{X^+}$, we shall say that the automaton D decodes F with delay of not more than t (with delay t) if $\exists t\geq 0$ such that

$$\forall a_{i_1}\ldots a_{i_k}\varepsilon A^+,\ \forall w_{i_1}\ldots w_{i_k}\varepsilon F(a_{i_1}\ldots a_{i_k}) \text{ and } \forall \beta\varepsilon C^t\ \ \phi(s_o,w_{i_1}\ldots w_{i_k}\beta)\geq a_{i_1}\ldots a_{i_k} \tag{3}$$

In such a case the automaton D is called a decoding automaton (or simply a decoder). The smallest number t, τ_D, such that (3) holds will be called the delay (of decoding) of the decoding automaton D. One has that an automaton decodes F iff it is an inverse machine for the nondeterministic gsm M that implements F.

Lemma 4 |3|. Let $M=(Q,q_o,A,X,\lambda,\delta)$ the nondeterministic gsm that implements F and $M^{-1}=(S,s_o,X,A,f,\phi)$ a deterministic finite automaton. The serial connection $M^{-1}M$ of M and M^{-1} in its initial state is a delay machine with (maximum) delay P iff M^{-1} is a decoding automaton with delay P for F.

■

Corollary 2 . A multivalued encoding F has finite decipherability delay P iff P is the smallest delay of all automata that decode F.

■

Theorem 4 . Let F be a multivalued encoding. A necessary and sufficient condition for a decoding automaton to exists for F (for F to have finite decipherability delay) is that $\exists\, t\geq 0$

$\forall\, a_{i_1}\ldots a_{i_k}\ \varepsilon A^+,\ \forall\, w_{i_1}\ldots w_{i_k}\ \varepsilon F(a_{i_1}\ldots a_{i_k})\ \ \forall\, \beta\varepsilon C^t$ there takes place

$$G(w_{i_1}\dots w_{i_K}\beta) \geq a_{i_1}\dots a_{i_K}\,. \tag{4}$$

The smallest number t, τ, such that (4) holds is equal to decipherability delay P of F.

Proof. Let $D=(S,s_o,X,A,f,\phi)$ be a decoding automaton for F with delay t_o. One has that

$$\forall\ \beta\varepsilon B(C^+) \quad G(\beta) \geq \phi(s_o,\beta). \tag{5}$$

Indeed assume that (5) is not true: that is, $\exists\ \beta\varepsilon B(C^+)$ such that $\phi(s_o,\beta)=a_1a_2\dots a_h\dots a_k$, $G(\beta)=a_1a_2\dots a_h$, $0 \leq h < k$. Then it is possible to distinguish the two following situations

a) there exists an initial decomposition of β, $\delta(\beta)=w_{i_1}w_{i_2}\dots w_{i_h}w_{i_{h+1}}\dots w_{i_m}\overline{\beta}$, $n \geq h$, $\overline{\beta}\varepsilon P(C)$ such that $w_{i_1}w_{i_2}\dots w_{i_h}\varepsilon F(a_1a_2\dots a_h)$, $w_{i_{h+j}}\varepsilon F(a_{i_{h+j}})$, $1 \leq j \leq n-h$, $a_{i_{h+j}}\varepsilon\{A,\lambda\}$ and $a_{i_{h+1}}\neq a_{h+1}$. Hence, if γ is such that $\overline{\beta}\gamma\varepsilon C^{t_o}$, it results

$\phi(s_o,w_{i_1}w_{i_2}\dots w_{i_h}\dots w_{i_m}\overline{\beta}\gamma) \geq a_{i_1}a_{i_2}\dots a_{i_h}a_{i_{h+1}}\dots a_{i_m}=a_1a_2\dots a_ha_{i_{h+1}}\dots a_{i_m} \geq \phi(s_o,\beta) \geq$ $a_1a_2\dots a_ha_{h+1}\dots a_k$, that contradicts the assumption $a_{i_{h+1}}\neq a_{h+1}$;

b) there exists an initial decomposition of β, $\delta(\beta)=w_1w_2\dots w_{h-1}\overline{\beta}$, $w_1w_2\dots w_{h-1}\varepsilon F(a_1\dots a_{h-1})$ such that all codewords beginning with $\overline{\beta}$ belong to $F(a_h)$. One then gets that for $w\varepsilon F(a)$, $a\neq a_{h+1}$, and $\gamma\varepsilon C^{t_o}$ it results

$\phi(s_o,w_1\dots w_{h-1}w_hw\gamma)\geq a_1\dots a_{h-1}a_ha \geq \phi(s_o,\beta)\geq a_1\dots a_ha_{h+1}\dots a_k$ that contradicts the assumption $a\neq a_{h+1}$. From (5) and by definition of a decoding automaton one gets finally (4).

Assume (4). We shall provide an algorithm for constructing a decoding automaton D for F with minimum decoding delay τ that, by Corollary 2, coincides with the decipherability delay P of F. Let us consider the automaton $D=(S,s_o,X,A,f,\phi)$ defined as follows

$S=\{s_{(Y,Z)}\,|\,(Y,Z)\varepsilon M\}$, $s_o=s_{(\{\lambda\},\emptyset)}$, where $Y\subset B(C^+)$, $Z\subset T(C)$ and M constructed according to rules i and ii

i) $(\{\lambda\},\emptyset)\varepsilon M$

ii) For any $b\varepsilon X$, if $(Y,Z)\varepsilon M$ and $b^{-1}Z$, $Y\cdot b\cap B(C^+)$ are not both empty then $H(Y,Z,b)\varepsilon M$.

It is easy to see that M and therefore S is finite |3|. The function of transitions f and the output function ϕ, defined on the set of pairs $(s_{(Y,Z)},b)\varepsilon S\times X$ such that $b^{-1}Z$ and $Y\cdot b\cap B(C^+)$ are not both empty, are determined in the following way:

$$\phi(s_{(Y,Z)},b)=G(Y,Z,b) \tag{6}$$

$$f(s_{(Y,Z)},b)=s_{H(Y,Z,b)} \tag{7}$$

Notice that the above definitions are consistent. Indeed if $(s_{(Y,Z)},b)\varepsilon S\times X$ (6) gives $\phi(s_{(Y,Z)},b)\varepsilon A^*$. On the other hand, because of ii, $H(Y,Z,b)\varepsilon M$, i.e., $s_{H(Y,Z,b)}\varepsilon S$. We shall show that D is a decoder and has decoding delay τ. For that, because of (4), it is sufficient to show that

$$\forall\ \beta\varepsilon B(C^+) \quad \phi(s_o,\beta)=G(\beta). \tag{8}$$

We shall prove the above formula (8) simultaneously with formula

$$f(s_o,\beta)=s_{H(\beta)} \tag{9}$$

by induction on the length of word β. Let $l(\beta)=0$ (i.e. $\beta=\lambda$). By definition one gets $\phi(s_o,\lambda)==\lambda=G(\lambda)$, $f(s_o,\lambda)=s_o=s_{(\{\lambda\},\emptyset)}=s_{H(\lambda)}$. Let us now assume that (8) , (9) take place $\forall\ \beta\in B(C^+)$, $l(\beta)=n$. We shall prove that they take place also $\forall\ \bar{\beta}=\beta b\in B(C^+)$, $l(\beta)=n$ and $b\in X$, $l(\bar{\beta})=n+1$. By Lemma 3 $f(s_o,\bar{\beta})=f(s_o,\beta b)=f(f(s_o,\beta),b)=f(s_{H(\beta)},b)=s_{H(H(\beta),b)}=s_{H(\beta b)}$ and then (9). Moreover $\phi(s_o,\bar{\beta})=\phi(s_o,\beta b)=\phi(s_o,\beta)\phi(f(s_o,\beta),b)=G(\beta)\phi(s_{H(\beta)},b)=G(\beta)G(H(\beta),b)=G(\beta b)$. For (5) and (8) one gets that the decoding delay τ of the automaton built according to the method above described is P, the decoding delay of the encoding F.

It shuld be remarked that in the case of single valued codes the above decoders behave essentially as decoding automata that are usually considered for decoding ordinary codes (see for example |4|).

■

Example 2 .

Given $A=\{0,1\}$, $X=\{a,b,c,d\}$, $C=\{aa,aab,bb,bbc,cd\}$ the multivalued code $F(0)=\{aa,aab,bb\}$, $F(1)=\{bbc,cd\}$ has decipherability delay 1. The decoder is shown in Fig. 1.

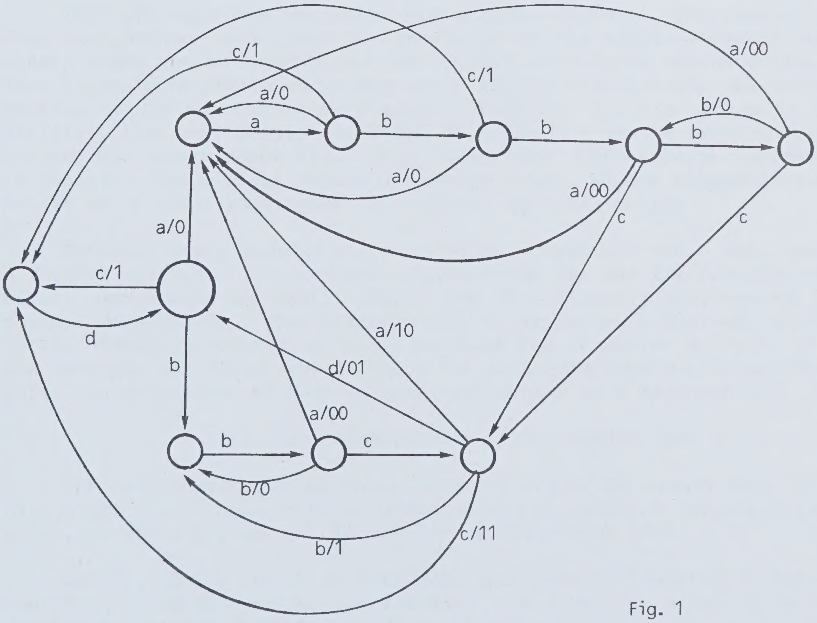


Fig. 1

REFERENCES

1) K. Sato, "A decision procedure for the unique decipherability of multivalued encodings," IEEE Trans. Inform. Theory, vol. IT-25, pp.356-360, 1979.

2) R.M. Capocelli, "A decision procedure for finite decipherability and synchronizability of multivalued encodings," IEEE Trans. Inform. Theory, vol. IT-28, pp. 307-318, 1982.

3) R.M. Capocelli and U. Vaccaro, "Structure of decoders for multivalued encodings," submitted.

4) V.I. Levenshtein, "Some properties of coding and self adjusting automata for decoding messages," Problemy Kyberneticki, vol.11, pp.63-121, 1964.

ERROR DETECTION, A GEOMETRIC APPROACH

R. Kreidieh WARD
Department of Electrical Engineering, University of British Columbia
Vancouver, B.C., Canada V6T 1W5

ABSTRACT

Error detection and correction is traditionally a highly mathematical subject. For parallel processing, the main applications in computer systems, this subject could be introduced in an intuitional geometrical fashion. The main tool is the Karnaugh map.

In this paper most of the basic and relevant theory will be re-introduced via this new approach. This will also lead into further novel results. Results known for SEC-DED (single-error correcting double error detecting) codes are found to be easily generalized to higher order t-error correcting (t+1) error detecting codes.

I. INTRODUCTION

Error codes in use at present in computer systems have been adapted from codes developed in the past for communication systems. The constraints in many computer applications though, differ from those in communications. Generally, information processing should be handled in parallel instead of serially, (as is done in communication systems) because the time allowed for coding and decoding is more crucial. A new approach which constructs codes with the above constraints as part of the design uses the Karnaugh map. This novel and simple approach extends to t-error correcting and detecting codes for all t. Results comparable in size to the best binary linear known codes are obtainable.

This approach has the following merits: firstly, the theory involved does not involve more than the knowledge of the simple Karnaugh maps and binary sums, yet it extends for any t-error correcting and detecting codes. Thus it could be presented to engineers and other scientists and technicians working on implementation with ease. Secondly, all the coding is done in parallel, i.e. each parity check bit is a function of the data bits only and not of any other check bit. This means that the hardware implementation is faster. Speed might not be a crucial matter in the communications area but it is of vital importance in computer applications /4.

Thirdly, every code is represented in a Karnaugh map. This consitutes a further merit for the hardware engineer who can use the Karnaugh map as a direct implementation tool. Codes are traditionally represented via the parity check matrix. The latter could be mapped on a Karnaugh map or vice versa. However, working with the Karnaugh map is easier since it is visual and one can use human intuition. The following section illustrates this point via an example of Hamming code represented in a Karnaugh map.

II. THE KARNAUGH MAP OF THE HAMMING CODE

For single-error correction, assume a string of binary data of eleven bit length (x1,x2,x3...x11) is coded into the codeword (x1,x2,x3...x11,p1, p2,p3,p4) where p1, p2, p3, p4 are four parity-check bits.

Let S1, S2, S3 and S4 be four sets such that Si contains pi and some of the Xi's; and let pj be sum function mod 2 of the elements in Sj (all arithmetic in this paper is mod 2) as follows:

$$\begin{array}{ll} S1 = \{x1,x3,x5,x7,x9,x11,x8,p1\} & p1 = x1 + x3 + x5 + x7 + x9 + x11 + x8 \\ S2 = \{x1,x2,x5,x6,x9,x10,x8,p2\} & p2 = x1 + x2 + x5 + x6 + x9 + x10 + x8 \\ S3 = \{x1,x2,x3,x4,x9,x10,x11,p3\} & p3 = \ldots \\ S4 = \{x1,x2,x3,x4,x5,x6,x7,p4\} & p4 = \ldots \end{array}$$

For single-error correction, only "one single error" is assumed to occur in the codeword. The question is which of the x's should belong to each set si so that if one single error occurred, say, in one of the x's or the p's, then the code could point to the erroneous bit. This pointer in the literature is called the "syndrome". Each of the bits which could go wrong should be marked by a unique marker or pointer.

Assuming that during transmission a bit, x7 say, went wrong. Since x7 lies in S1 and S4, at the receiver the sum of the elements in Si would be equal to 1 for i = 1,4 and 0 for i = 2,3. Thus, the syndrome is [1,0,0,1]. This syndrome corresponds to x7 thus x7 is to be corrected. If the syndrome is [0,0,0,0], this means no error has occurred. If a check bit, say p3 went wrong during transmission, then at the receiver the sum of elements in S3 would only equal 1. The syndrome is thus [0,0,1,0]. This corresponds to p3 and thus p3 is in error.

To find unique syndromes for each bit, the Karnaugh map is used. The 4 sets si's could be represented as an "empty" Karnaugh map. For example in Fig. 1, the set s1 is comprised of a total 8 squares, those in the second and 3rd rows. The set s4 are all the squares in the 3rd and 4th columns, etc.

Since p1 is to belong to set s1 only, it will be placed in the corresponding square s1, $\overline{s2}$, $\overline{s3}$, $\overline{s4}$. Similarly, so will p2, p3 and p4. The null square which does not belong to any set is named N.

The data bits, the x's, are then placed in any fashion in the squares of the map so that each bit uniquely occupies one square, as in Fig. 1a. Figure 1a now represents a "single error correcting" code. This code has p1, x8, x5, x7, x11, x9, x1 and x3 in set S1 and p2, x6, x8, x5, x9, x1, x10 and x2 in S2, and so on.

The code of Fig. 1a is represented in the literature by the following Hamming parity check matrix:

$$H = \begin{bmatrix} 1&0&1&0&1&0&1&1&1&0&1&1&0&0&0 \\ 1&1&0&0&1&1&0&1&1&1&0&0&1&0&0 \\ 1&1&1&1&0&0&0&0&1&1&1&0&0&1&0 \\ 1&1&1&1&1&1&1&0&0&0&0&0&0&0&1 \end{bmatrix}$$

Notice that each column of the matrix correponds to a syndrome or a square.

The code of Fig. 1 is a "single error" correcting code. For higher order error correction and detection more parity check bits will be needed. It is easily seen that if c is a codeword generated as explained above then Hc=0.

The ideas from the above example for single error correction could be extended to higher order error correction and detection. For a binary word, if certain occurrences of errors are possible, say, in only (x1,x5) and (x2,x3), then to be able to correct these errors, each of these possible error occurrences should have its own unique syndrome or its own unique square in the map. This is for error correction. For error detection more than one possible error combination could share the same square. Thus the problem is that of finding or allocating the right square for each error possibility so that these errors are detected and or corrected.

If two bits B_u and B_v were allocated the squares whose addresses were u and v respectively then the square for the double error occurrence of B_u and B_v will be automatically allocted and will have the address (u+v). This will be formulated as theorem 1 and it will be needed for the extension of earlier results on single error correction to results on higher order error detection and correction.

Theorem 1

Assume two bits each having its own single error syndrome. The syndrome of the double error occurrence of the two bits is the sum (mod 2) of their single syndromes.

Proof

Assume two bits B1 and B2. B could be x or p. B1 belongs to the sets (S_{i1}, S_{i2}, ..., S_{iN}) and B2 belongs to the sets (S_{j1}, S_{j2}, ..., S_{jM}). If B1 is the only wrong bit then at the decoder (the receiving end) the calculated values for (p_{i1}, p_{i2}, ... p_{iN}) would be changed. Similarly, if only B2 went wrong, then (p_{j1}, p_{j2}, ..., P_{jM}) would be changed. If both B1 and B2 went wrong then (p_{i1}, p_{i2}, ..., p_{in}, p_{j1}, p_{j2}, ..., p_{jM}) would be changed if the p_i and the p_j subsets do not overlap. If these subsets overlap then for those p's occurring in both subsets the 2 changes would cancel out. The result would be the (mod 2) sum of both subsets which is the (mod 2) sum of the single syndromes.

Q.E.D.

Definition

The (Hamming) distance between any two words of equal length is defined as the number of bits by which they differ.

Definition

The (Hamming) weight of a binary word is the number of 1's in the word.

Definition

If two squares differ by only the one set S_i then one square is said to be the image of the other with respect to the S_i axis.

Notice that in the single error correcting map of Fig. 1 the squares of any two bits have a minimum distance of one.

III. ERROR DETECTION

Hsiao [2] has introduced a class of optimal single-error correcting double error detecting codes which provide faster and simpler hardware implementation. In what follows first these codes will be deduced via the Karnaugh map. These then will be generalized for t error correction (t+1) error detection.

Consider any single error correcting code, e.g., that of Fig. 1. By adding another parity bit, p5, s.t. p5 = x1 + ... x11 + p1 + ... + p4 the code becomes SEC-DED (single-error correcting and double-error detecting). In the Hamming matrix $\overline{/^3}$ another row of 1's is added and also another column $[00001]^t$. Now the set S5 contains all the x's and all the p's. Since the newly added p5 is a function of the other parity-check bits, this is serial coding, and the circuit has to wait for p1,p2,p3,p4 to be computed before it computes p5. It is desired that each parity bit, including p5, be a function of the data bits only so as to minimize encoding and decoding time. It is also desired that the maximum time taken to obtain the p_i's is minimal.

The Karnaugh map of the extended Hamming code with the five parity check bits is represented in Fig. 2. This is the image of Fig. 1 with respect to the S5 axis. If the syndrome lies in S5, the single bit corresponding to that square is corrected; and if it lies in $\overline{S5}$ there is a double error somewhere, except if it lies in the N square, then no error is present.

If each of the parity bits is to be a function of the data bits only, then one could see that p1,p2,p3 and p4 should be placed back in their positions as in Fig. 1, that is, p_i lies in the intersection of S_i and $\overline{S}_k$ for all $K \neq i$.

A method obtained for correcting t-error and detecting any (t+1) errors from a t-error correcting code that will minimize circuit time delay is presented below. Here (by adding one extra parity check bit to), any given t-error correcting code will become a t-error correcting and a (t+1) error detecting code while keeping the property that each parity bit is a sum of data bits only. Before proving the general theorem some definitions and preliminary theorems are due.

Definition

A binary word will be defined as even, if its Hamming weight (the sum of the non-zero bits) is even; and odd, if the Hamming weight is odd.

All sums in this paper refer to exclusive-or operations or (mod 2) sums.

Theorem 2

The sum of any two odd words results in an even word.

Proof

Suppose the two odd words are a and b, where a has the odd weight r and b has the odd weight s. b is the sum of connonical row vectors, $b_j (j = 1,s)$, where each vector b_j has only one non-zero entry; and the sum of all the b_j's add to b. From this we get the following:

$$a + b = a + b_1 + b_2 + \dots + b_s$$

$a + b_1$ results in an even vector, since a has an odd number of 1's; and b_1 has only 1. The even vector $a + b_1$ added to the odd vector b_2 results in an odd vector, etc. Since s is an odd number, the resulting vector a + b would be even.

Q.E.D.

Corollary 1

The sum of an odd word and an even word is an odd word.

Corollary 2

The sum of any two even words is an even word.

The proofs for corollaries 1 and 2 are very similar to that of Theorem 2 and thus shall be omitted.

In a Karnaugh map each square is represented by a binary vector. The square is defined as even if its binary word representation (addresses) is even, and odd otherwise. For example, in Fig. 1, the square where x7 lies is [1,0,0,1] corresponding to $\{S1, \overline{S2}, \overline{S3}, S4\}$; thus this square is an even square. In any Karnaugh map exactly half the squares are even and the other half are odd. Fig. 3 shows the odd and even squares in a Karnaugh map of order 5, the shaded squares being the even ones.

Now, assuming one starts with any single-error correcting code, e.g. the Hamming code shown in Fig. 1, and adds one more parity bit, the number of squares in the map becomes twice its original size. To make the above code double-error detecting as well, one proceeds as follows. Move bit positions which originally lay in an even square to their images around the S5 axis. That is, such bits would belong to S5 as well as their original sets. Thus all bits now lie in odd squares. Since every bit would still lie in a square not shared by any other bit, the code is still single-error correcting. If two errors occurred, then the syndrome would be even, since from Theorems 1 and 2 the sum of any two odd syndromes is even. Thus, if the syndrome corresponds to an even square, a double error has occurred.

The parity check matrix of Fig. 4 is as follows:

$$\begin{bmatrix}
1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

It could be shown that this code is optimal in terms of time; i.e., it results in minimum encoding and decoding time. Observe that the single error correcting code has been extended to a SEC-DED code with no addition in delay execution time. That is, the delay time taken in encoding or decoding the new code (of Fig. 4) is exactly the same as that of the original single-error correcting code from which the new code has been obtained. The delay time is governed by the time taken to add (exclusive-or) the maximum number of bits in any set. In both cases (Figs. 1 and 4) it is 8 bits. The maximum number of bits belonging to any set is the same as the maximum number of 1's in any row in the matrix.

For the general case, it could be shown (theorem 5) that to extend a t-error correcting code to t-error correcting and (t+1) error detecting code with no additions in delay time the following sufficient condition should be satisfied: for the original t-error correcting code the total number of even data bits should be less than the maximum number of bits in any set of the code.

The above extension of the code to double error detection could be reversed. That is, given a (SEC-DED) code, constructed by adding another parity bit as explained above, then by eliminating the last parity bit, one gets back again a single error correcting code. Actually, this case is of a more general nature, for it could be shown that by discarding any of the parity bits (not necessarily the last one), the resulting code becomes single-error correcting. This is generalized in Theorem 4. One can see this fact directly from Fig. 4. For example, assume S2 is removed, then p2 is discarded: x5 will fall in the square between p4 and x3, which is the image of x5 with respect to the S2 axis, similarly, x9 will fall on its S2 image, which is the square between p1 and p3, and x2 will fall on its S2 image, which is between x3 and x4, etc. The resulting map will be single error correcting since each square is occupied by only one position for each syndrome. In the corresponding parity-check matrix, this means discarding any one of the rows and the remaining zero column.

The reversal of the above extension of the single-error correcting to SEC-DED could be generalized to any t-error correcting codes.

The general theorems for error detecting are as follows.

Theorem 3

With any t-error-correcting code, by adding one extra parity bit and moving the positions of all single-errors originally occupying an even square in the Karnaugh map to their image squares around the last added axis, the code resulting is a t-error-correcting (t+1) error detecting code.

In matrix notation, this means adding another row to the parity check matrix and another column. The entries of the row to be added would be 1 if the column above it is even and 0 otherwise. The column would have 1 in the (t+1) row, and 0 otherwise.

Corollary

For a t-error correcting code, if all the data bits occupy odd squares, then the code is t-error correcting and (t+1) error detecting as well.

Theorem 4

With any t-error correcting (t+1)-error detecting parallel code, whose single error syndromes are odd, by discarding any of its parity bits, the resulting code is a t-error correcting one.

In matrix notation this means that in the parity check matrix, if any row is discarded (plus the remaining zero column), then the resulting matrix is a parity check matrix for a t-error correcting code.

If a code is to correct double-errors as well as single errors, then every data bit x_i , every parity-check bit p_i, and every double-error $x_i x_j$, $x_k p_\ell$ and $p_m p_n$ should have a unique square location in the map. Fig. 5 is such a map.

To construct such a map, first place all the check bits p_i's in their squares, then place all the double occurrences of $p_i p_j$, $V_{i,j}$. Now the map is ready for placing data bits in squares. To do so, search for an empty square for the first-data bit x1 such that all occurrences of $x1p_i$, Vi are empty. Once such a square is found, assign x1 to that square and place all $x1p_i$, Vi in their respective squares. After placing any data bit, repeat the following; search for an eligible empty square for the next data bit x_j such that the corresponding squares for x_j and all the $x_j x_i$ for all previously assigned x_i's are empty. If such a square is not found, stop. Otherwise place x_j and all its corresponding double errors $x_j p_i$ and $x_j x_i$ in their respective squares and repeat.

For minimum time delay hardware implementation, the following theorems serve as quidelines for the design of such codes. These theorems shall only be stated here.

Theorem 5

It is possible, by adding one extra parity bit, to extend a t-error correcting code to a t-error correcting (t+1) error detecting code, with no addition in the maximum number of bits to be summed (exclusive-ored), if the total number of even data bits in the t-error correcting code is less than the maximum number of bits in any of its sets.

Theorem 6

For a t-error correcting code with n data bits and p parity check bits, the lower bound on the maximum number of bits to be decoded in obtaining any one bit of the syndrome is:

$$= \frac{(nx2t)}{p} + 1 \quad , \quad \text{if } \frac{nx2t}{p} \text{ has no remainder}$$

$$= \frac{nx2t}{p} + 2 \quad , \quad \text{if } \frac{nx2t}{p} \text{ has a remainder}$$

Corollary

For a t-error correcting (t+1) error detecting code with n data bits and p parity check bits, the lower bound on the maximum number of bits to be decoded in obtaining any bit of the syndrome is:

$$= \frac{n(2t+1)}{p} + 1 \text{ if } n(2t+1) \text{ is divisible by } p$$

$$= \frac{n(2t+1)}{p} + 2 \text{ otherwise}$$

IV. SUMMARY

A helpful tool to construct error correcting and detecting codes is the Karnaugh map. Such an approach has an inherent characteristic leading to codes with parallel hardware implementation. It is thus better suited for computer and other parallel processing applications. The mathematics involved is simple and intuitional.

Original results on error detection were presented. Although only single-error correction have been discussed via this approach, codes for higher error correction and/or with desired specifications could be constructed. Also, conditions for such codes which lead to minimum hardware time execution could be derived.

REFERENCES

1. R.W. Hamming, "Error Detecting and Error Correcting Codes", Bell System Technical Journal, vol. 29, pp. 147-160.

2. M.Y. Hsiao, "A Class of Optimal Minimum Odd-weight-Column SEC-DED Codes", IBM Journal of Research and Development, vol. 14, No. 5, July, 1970, pp. 395-401.

3. I.J. MacWilliams and N.J.A. Sloane, "The Theory of Error-Coding Codes", Part I, North-Holland Publishing Co., 1977.

4. D.K. Pradham and J.J. Stiffer, "Error-Correcting Codes and Self-Checking Circuits", COMPUTER, IEEE, March, 1980.

5. J. Wakerly, "Error Detecting Codes, Self-Checking Circuits and Applicators", North-Holland, New York, Computer Design and Architecture Series, 1978, Chapter 9.

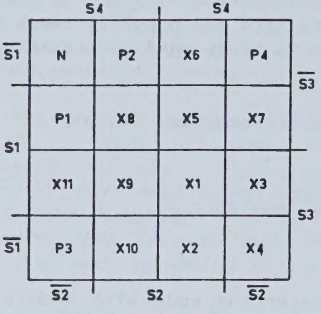

$\overline{S4}$ S4
$\overline{S1}$ N P2 X6 P4
$\overline{S3}$
P1 X8 X5 X7
S1
X11 X9 X1 X3
S3
$\overline{S1}$ P3 X10 X2 X4
$\overline{S2}$ S2 $\overline{S2}$

Fig. 1

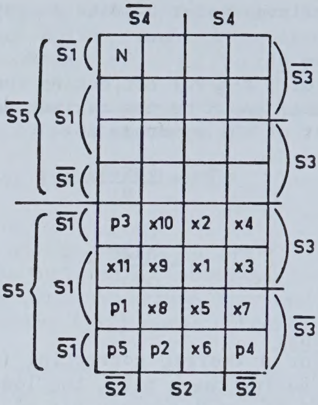

$\overline{S4}$ S4
$\overline{S1}$ N
$\overline{S3}$
$\overline{S5}$ S1
S3
$\overline{S1}$
$\overline{S1}$ p3 x10 x2 x4
S3
x11 x9 x1 x3
S5 S1
p1 x8 x5 x7
$\overline{S3}$
$\overline{S1}$ p5 p2 x6 p4
$\overline{S2}$ S2 $\overline{S2}$

Fig. 2 Single-Error Correction Double-Error Detection by Extending the Hamming Code.

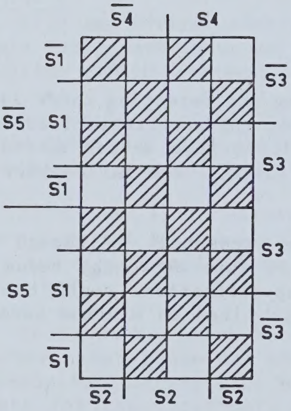

$\overline{S4}$ S4
$\overline{S1}$
$\overline{S3}$
$\overline{S5}$ S1
S3
$\overline{S1}$
S3
S5 S1
$\overline{S3}$
$\overline{S1}$
$\overline{S2}$ S2 $\overline{S2}$

Fig. 3 Shaded Squares are the Even Squares.

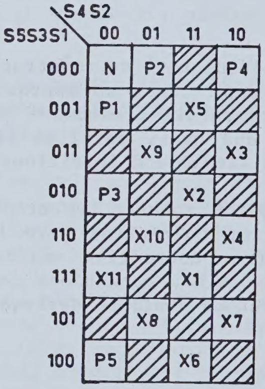

S4S2
S5S3S1
00 01 11 10
000 N P2 P4
001 P1 X5
011 X9 X3
010 P3 X2
110 X10 X4
111 X11 X1
101 X8 X7
100 P5 X6

Fig. 4 Single-Error Correction Double-Error Detection with Independent Parity Bits.

S6S4S2 S7S5S3S1	000	001	011	010	110	111	101	100
0000	N	P2	P2P4	P4	P4P6	X4P7	P2P6	P6
0001	P1	P1P2	X1P3	P1P4	X2X3			P1P6
0011	P1P3	X1P4	X1	X1P2		X1P6		
0010	P3	P2P3	X1P1	P3P4				P3P6
0110	P3P5		X3X4					X3P7
0111			X1P5	X2P7			X2X4	
0101	P1P5							
0100	P5	P2P5		P4P5				P5P6
1100	P5P7	X1X2				X4P5		X3P3
1101				X2P3		X1X3		
1111	X2P4		X2P2	X2	X2P6			X3P1
1110	X3P6			X2P1	X3P4		X3P2	X3
1010	P3P7					X4P3		X3P5
1011			X1P7	X2P5				X1X4
1001	P1P7					X4P1		
1000	P7	P2P7	X4P6	P4P7	X4P2	X4	X4P4	P6P7

Fig. 5 Karnaugh Map for a Double-Error Correcting Code.

EVALUATION OF STEADY-STATE RESPONSES OF MILDLY NONLINEAR SYSTEMS

I. W. SANDBERG
Bell Laboratories, Murray Hill, New Jersey 07974

ABSTRACT

In this paper a theorem is given that provides conditions for the existence of, and recursive relations for evaluating, the steady-state response of a certain general type of "mildly nonlinear" time-invariant system with inputs that are asymptotically almost periodic.

1. INTRODUCTION AND SUMMARY

In this paper we build on results and ideas in [1], and give a theorem that provides conditions for the existence of, and recursive relations for evaluating, the steady-state response of a certain general type of "mildly nonlinear" time-invariant system. The response is given in the form of a locally convergent series expansion in which the terms of different order are almost periodic functions.

In work prior to [1] and this paper on similar steady-state response evaluation problems (see, for example, [2]-[4]), corresponding questions concerning existence, representation, interpretation (see [1, Section 1.1]) and convergence are not seriously addressed, nor has the theory of almost periodic functions been used. Typically, authors have assumed the existence and appropriateness of certain Volterra series representations, and have used multi-variable Fourier transform techniques to give expressions for the Fourier coefficients of the response which is taken to be represented by a suitable convergent series.

With regard to the present paper, one point of particular interest is that while functional expansion techniques play a central role in the proofs, Volterra series methods (i.e., representations involving iterated integrals) are not needed or used. Another is that we are able to show that the systems we consider possess a certain *preservation* property described below which implies that for the large class of inputs addressed by our theorem, the Fourier series for the steady-state response actually converges to the response for *every* value of the time variable t. (The series for a general almost periodic function converges with respect to the usual metric associated with the space (see [5, p. 255] and (iii) of Section 2.3); this convergence is very different from pointwise convergence, even though almost periodic functions are uniformly continuous. In fact, we remind the reader that there are [6, p. 115] continuous *purely periodic* functions of t for which the corresponding Fourier series does not converge for some values of t.)

1.1 The Model and Background Material

As in [1] and in earlier work (see, for example, [7], [8]) attention is focused here on a system model characterized by five operators: a nonlinear operator N and linear operators A, B, C and D.

In this model (see [7] for additional details)* the system input v and corresponding output w are related by the equations

$$y = Nx \tag{1}$$

$$x = Av + Cy \tag{2}$$

$$w = Dv + By \tag{3}$$

where x and y, respectively, can be interpreted as the input and output to the nonlinear portion of the system. In (1), (2), and (3), y, x, v, and w are n-vector valued and defined for $t \geqslant 0$ where, as usual n is an arbitrary positive integer. The equations of a very large class of systems can be written in the form (1)-(3), with N memoryless. It is of course not necessary that the equations of the system to be studied be given *at the outset* in the form (1)-(3); see [7, Appendices I and II]. For the moment, and to simplify the discussion in this section and in Section 1.2, let $n = 1$.

In [8] results are given which show that, under certain reasonable conditions on the five operators, there is a positive constant δ such that for all piecewise continuous (or even merely Lebesgue measurable) complex bounded inputs v with $|v(t)| < \delta$ for $t \geqslant 0$ the equations (1), (2), and (3) have a solution that is unique in basically a standard local sense, and w has the representation

$$w(t) = \sum_{m=1}^{\infty} \int_0^t \cdots \int_0^t h_m(t-\tau_1,\ldots,t-\tau_m) v(\tau_1) \ldots v(\tau_m) \, d\tau_1 \ldots d\tau_m \,, \; t \geqslant 0 \tag{4}$$

(in the sense of uniform convergence on $[0,\infty)$), in which each h_m in the Volterra series (4) is a bounded function that satisfies the integrability condition

$$\int_{[0,\infty)^m} |h_m(\tau_1,\ldots,\tau_m)| \, d(\tau_1,\ldots,\tau_m) < \infty \, .$$

The input-output relation (4) is clearly time-invariant in the usual sense. In [7] related results are given for time-varying systems, i.e. for systems that are not necessarily time invariant. The results obtained there by specializing to the time-invariant case involve weaker assumptions than those in [8] concerning N and the linear maps A, B, C, and D. These weaker assumptions allow consideration of certain very important cases, and they lead to a more complicated expression [7, Theorem 2] for the mth term in the series expansion for w. Both [7] and [8] treat the case in which n is not necessarily equal to unity.

Consider once more the special case (4). In [1] it is proved that the bound δ mentioned above can be taken so that the *associated* expansion

---

* Assumptions not explicitly made in this paper are not needed to prove our results. However, for the purpose of providing a setting for the *interpretation* of our results it is assumed that the typically very weak hypotheses concerning the range and domain of N, A, B, C, and D of [7, Section 2.2] hold.

$$y(t) = \sum_{m=1}^{\infty} \int_{-\infty}^{t} \cdots \int_{-\infty}^{t} h_m(t-\tau_1,\ldots,t-\tau_m)x(\tau_1)\ldots x(\tau_m)d\tau_1\ldots d\tau_m \ , -\infty < t < \infty \tag{5}$$

converges uniformly in t when $|x(t)| < \delta$ for all t (and x is Lebesgue measurable), that this associated expansion has certain fundamental properties concerning almost periodic functions and their spectra, and that it is related via basic propositions concerning asymptotics in the variable t to the original expansion (4). These properties and propositions show (among other things), that under the conditions assumed, an almost periodic steady-state response exists and can be evaluated using (5) when, in addition to being bounded as indicated above, v is the restriction to $t \geqslant 0$ of a sum of an almost periodic function x with $|x(t)| < \delta$ for all t and a bounded Lebesgue measurable function that approaches zero as $t \to \infty$. Corresponding results in a more general setting regarding the five operators are given also, and these more general results address the $n \geqslant 1$ case (see [1] for the details as well as for a discussion of the relation of the results to earlier work, and for additional related references of a general nature).

1.2 Description of Our Main Result

The Fourier coefficients and exponents of the steady-state response are typically the items of interest. While these can be evaluated using (5), or its more general counterpart (see [1, Theorem 7]) that corresponds to weaker integral operator representation hypotheses on A, B, C, and D, and to weaker conditions on N, we do not proceed in that way. It will become clear in Section 2 of the full-length version of this paper (Section 2 is omitted in this version) that we take a more general approach based on functional expansion techniques, but *not* on Volterra series methods. The approach enables more ground to be covered with regard to the range of applicability of our results, and provides a much better understanding of the setting in which the type of problem addressed can be solved.

We now turn to a brief and simplified description of our main result, which is Theorem 3 in Section 2.4.2 of the full paper. We being with a definition.

Let (AP) denote the set of all complex-valued functions q defined on $(-\infty,\infty)$ that can be expressed in the form

$$q(t) = \sum_{j=1}^{\infty} c_j e^{i\omega_j t}, \quad -\infty < t < \infty \tag{6}$$

with each ω_j real, each c_j a complex scalar, and with

$$\sum_{j=1}^{\infty} |c_j| < \infty .$$

(The motivation for using "(AP)" is simply that the set we have defined is a subset of the set of scalar-valued almost periodic functions.)

Suppose that N is memoryless and is represented as follows: $(Nu)(t) = \eta[u(t)]$ for each u defined on $t \geqslant 0$, where η takes complex numbers into complex numbers. Assume that $\eta(0) = 0$, and that η is analytic at the origin. Let $M(\ell)$ denote $\partial\eta^{\ell}(0)/\partial z^{\ell}$ for $\ell = 1,2,\ldots$. Suppose that A, B, C, and D are convolution operators with kernels a, b, c, and d, respectively, that vanish for negative arguments and are absolutely integrable on $[0,\infty)$. Let $\hat{A}(\omega)$ denote the transfer function

$$\int_0^{\infty} a(t)e^{-i\omega t}\,dt, \quad -\infty < \omega < \infty$$

and let $\hat{B}(\omega)$, $\hat{C}(\omega)$, and $\hat{D}(\omega)$ stand for the corresponding transfer functions associated with b, c, and d, respectively. Assume that the feedback part of the system represented by (1), (2), and (3), when linearized, is stable in the (very reasonable) sense of [9, p. 210]. This assumption implies [8, Proposition 7] that $\hat{C}(\omega)M(1) \neq 1$ $(-\infty < \omega < \infty)$. Let $\hat{E}(\omega)$ denote $[1 - \hat{C}(\omega)M(1)]^{-1}$ for all real ω.

Now let $G_1,G_2,\ldots$ and $K_1,K_2,\ldots$ be defined recursively as follows, where $c_1,c_2,\ldots$ and $\omega_1,\omega_2,\ldots$ are arbitrary complex numbers and real numbers, respectively:

$$G_1(c_1,\omega_1) = \hat{E}(\omega_1)\hat{A}(\omega_1)c_1$$

$$G_m(c_1,\ldots,c_m,\omega_1,\ldots,\omega_m) = \hat{E}(\omega_1+\omega_2+\ldots+\omega_m)\hat{C}(\omega_1+\omega_2+\ldots+\omega_m)\sum_{\ell=2}^{m}(\ell!)^{-1}\sum_{\substack{k_1+\ldots+k_\ell=m\\ k_j>0}}$$

$$M(\ell)G_{k_1}(c_1,\ldots,c_{k_1},\omega_1,\ldots,\omega_{k_1})\ldots G_{k_\ell}(c_{(m-k_\ell+1)},\ldots,c_m,\omega_{(m-k_\ell+1)},\ldots,\omega_m)$$

for $m \geqslant 2$, and

$$K_m(c_1,\ldots,c_m,\omega_1,\ldots,\omega_m) = \hat{B}(\omega_1+\omega_2+\ldots+\omega_m)$$

$$\sum_{\ell=1}^{m}(\ell!)^{-1}\sum_{\substack{k_1+\ldots+k_\ell=m\\ k_j>0}} M(\ell)G_{k_1}(c_1,\ldots,c_{k_1},\omega_1,\ldots,\omega_{k_1})\ldots G_{k_\ell}$$

$$(c_{(m-k_\ell+1)},\ldots,c_m,\omega_{(m-k_\ell+1)},\ldots,\omega_m)$$

for $m \geqslant 1$.

According to Theorem 3, there are real numbers $\delta > 0$ and r_m $(m = 1,2,\ldots)$ with

$$\sum_{m=1}^{\infty} r_m\delta^m < \infty$$

such that:

(i) For each $q \in (AP)$ with $q(t) = \sum_{j=1}^{\infty} c_j e^{i\omega_j t} (-\infty < t < \infty)$ and $\sum_{j=1}^{\infty} |c_j| < \delta$, and for each $m = 1,2,\ldots$,

$$\sum_{j=1}^{\infty} \hat{D}(\omega_j) c_j e^{i\omega_j t}$$

and

$$\sum_{j_1=1}^{\infty} \cdots \sum_{j_m=1}^{\infty} K_m(c_{j_1},\ldots,c_{j_m},\omega_{j_1},\ldots\omega_{j_m})\exp[i(\omega_{j_1}+\ldots+\omega_{j_m})t$$

converge uniformly in t on the entire interval $(-\infty,\infty)$ to elements $F_0 q$ and $F_m q$ of (AP), and the series

$$Fq(t) \triangleq \sum_{m=0}^{\infty} (F_m q)(t), \quad -\infty < t < \infty$$

converges uniformly in t to an element of (AP) with

$$\sum_{j=1}^{\infty} |\tilde{c}_j| \leqslant \sum_{m=1}^{\infty} r_m \left[\sum_{j=1}^{\infty} |c_j| \right]^m ,$$

where the $\tilde{c}_j$ are the Fourier coefficients of Fq.

(ii) For each bounded Lebesgue measurable v in (2) and (3) of the form $v(t) = q(t)+r(t)$ $(t \geqslant 0)$, in which $q \in (AP)$ is as in (i), $|q(t)+r(t)| < \delta$ $(t \geqslant 0)$, and $r(t) \to 0$ as $t \to \infty$, there is a solution x, y, and w of the system equations that is unique (in a very reasonable local sense specified by Theorem 3), and the output w satisfies

$$w(t)-(Fq)(t) \to 0 \quad \text{as} \quad t \to \infty$$

(i.e., w approaches the steady-state response Fq as $t \to \infty$).

Note that under the conditions described, the system (1)-(3) has the basic preservation property that $Fq \in (AP)$ whenever v is the restriction to $t \geqslant 0$ of a function belonging to (AP). In applications v is very often of the form just mentioned. In fact, typically only a finite number of the c_j are nonzero. Thus, a fundamental property of a large class of nonlinear systems is that their steady-state responses belong to (AP); this is of course a much stronger statement than simply that their responses are almost periodic.

Notice also that the K_m are linear in each c_j $(1 \leqslant j \leqslant m)$, and that each $F_m q$ is of degree m in the sense that $F_m \beta q = \beta^m F_m q$ for any complex scalar β.

The main difference between the result described above and Theorem 3 in the full paper is that much weaker hypotheses are used concerning A, B, C, and D. Also, the general n case is treated (cases where $n > 1$ are very often of interest), and some information concerning the r_m is given.

1.3 Outline of the Omitted Section 2

Section 2 begins with mathematical preliminaries, and a set $P_1[m]$ of multilinear maps with "decaying memory t_0-associates" which plays a key role in the paper is introduced. Two theorems are described which concern the existence and properties of a certain associate $\mathscr{E}\{M\}$ of an arbitrary element M of $P_1[m]$. These theorems are used later in the proof of our main result. The approach described can be used also to obtain results concerning the response of systems to other classes of inputs, such as certain bounded periodic inputs that are not continuous.

It is mentioned in Section 2 that the existence of certain Fréchet derivatives in L_∞ (more precisely $L_\infty(0,\infty)$) plays an important role in our proof and in earlier related work. A result of peripheral interest given in an appendix (included below) shows that for all but a trivial case corresponding derivatives *fail* to exist in L_p $(1 \leqslant p < \infty)$.

APPENDIX: The Lack of Existence of Certain F-Derivatives

In this appendix, $p \in [1,\infty)$ is arbitrary, L_p stands for the familiar Banach space of all Lebesque measurable functions f from $[0,\infty)$ into the complex numbers such that

$$_p\|f\| \triangleq \left(\int_0^\infty |f(t)|^p dt\right)^{\frac{1}{p}} < \infty ,$$

and $\theta_{(p)}$ is used to denote the zero element of L_p.

Let S be an open subset of L_p containing $\theta_{(p)}$, and let $F: S \rightarrow L_p$ be defined by $(Fx)(t) = g[x(t)]$, $t \geqslant 0$ for $x \in L_p$, where g takes the set of complex numbers into itself such that $g(0) = 0$. We shall prove the following:

Proposition. The Fréchet derivative $dF(\theta_{(p)})$ of F at the point $\theta_{(p)}$ exists if and only if g is linear.

Proof

Assume first that g is not linear, and suppose that $dF(\theta_{(p)})$ did exist. Then,

$$_p\|Fh - dF(\theta_{(p)})h\| = o(_p\|h\|) \text{ as } _p\|h\| \rightarrow 0 . \tag{47}$$

Let f be the element of L_p given by $f(t) = \beta$ $(0 \leqslant t \leqslant \epsilon)$ and $f(t) = 0$ $(t > \epsilon)$ in which β is a nonzero complex constant and ϵ is a positive number. Clearly, $_p\|f\| = |\beta|\epsilon^{\frac{1}{p}}$. By (47),

$$\left[\int_0^\infty |(Ff)(t)/_p\|f\| - [dF(\theta_{(p)})f](t)/_p\|f\||^p dt\right]^{\frac{1}{p}} \to 0$$

as $\epsilon \to 0$. In particular,

$$I(\beta,\epsilon) \triangleq \left[\frac{1}{\epsilon}\int_0^\epsilon \left|\frac{g(\beta)}{\beta} - [dF(\theta_{(p)})u_\epsilon](t)\right|^p dt\right]^{\frac{1}{p}} \to 0 \tag{48}$$

as $\epsilon \to 0$, where u_ϵ is f with β replaced with unity. Since g is assumed to be not linear there is a complex number β_1 such that $g(\beta_1) \neq [g(\beta)/\beta]\beta_1$. Clearly $\beta_1 \neq 0$, and (48) holds with β replaced with β_1. By Minkowski's inequality,

$$\left[\frac{1}{\epsilon}\int_0^\epsilon \left|\frac{g(\beta)}{\beta} - \frac{g(\beta_1)}{\beta_1}\right|^p dt\right]^{\frac{1}{p}} \leqslant I(\epsilon,\beta) + I(\epsilon,\beta_1)\ . \tag{49}$$

This is a contradiction, because the left side of (49) is nonzero and independent of ϵ, while the right side approaches zero as $\epsilon \to 0$. Thus, $dF(\theta_{(p)})$ fails to exist whenever g is not linear.

Since $dF(\theta_{(p)})$ obviously exists when g is linear, our proof is complete.

The proposition holds also if L_p is defined instead over the field of *real* scalars and/or the interval of definition $[0,\infty)$ of the functions is replaced with $(-\infty,\infty)$, $[a,b]$, $[a,\infty)$, or $(-\infty,b]$ in which $-\infty < a < b < \infty$.* It shows that there is *no* sense in which a nonlinear g can be "sufficiently smooth" that $dF(\theta_{(p)})$ exists, and is apparently counterintuitive. In fact, the literature contains some inappropriate suggestions and incorrect statements to the contrary, even though it is well known that convergence in L_p with $1 \leqslant p < \infty$ does not imply convergence in L_∞.

REFERENCES

1. I. W. Sandberg, "The Mathematical Foundations of Associated Expansions for Mildly Nonlinear Systems," IEEE Transactions on Circuits and Systems, Vol. 30, No. 7 (July 1983), pp. 441-455.

2. E. Bedrosian and S. O. Rice, "The Output Properties of Volterra Systems (Nonlinear Systems with Memory) Driven by Harmonic and Gaussian Inputs," Proc. IEEE, Vol. 59, No. 12 (December 1971), pp. 1688-1707.

3. S. Narayanan, "Application of Volterra Series to Intermodulation Distortion Analysis of Transistor Feedback Amplifiers," IEEE Transactions on Circuits and Systems, Vol. 17 (November 1970), pp. 518-527.

---

* The difficulty described above does not arise in connection with *Gateaux* derivatives [10, p. 657] which are weak functional derivatives related to the directional derivatives of classical calculus.

4. A. A. M. Saleh, "Matrix Analysis of Mildly Nonlinear Multiple-Input, Multiple-Output Systems with Memory," Bell Syst. Tech. J. Vol. 61 (9), November 1982, pp. 2221-2243.

5. F. Riesz and B. Sz-Nagy, *Functional Analysis*, New York: Frederick Ungar Publishing Co. (1955).

6. I. J. Maddox, *Elements of Functional Analysis*, London: Cambridge University Press (1970).

7. I. W. Sandberg, "On Volterra Expansions for Time-Varying Nonlinear Systems," IEEE Transactions on Circuits and Systems, Vol. 30, No. 2 (February 1983), pp. 61-67.

8. __________ , "Expansions for Nonlinear Systems," Bell Syst. Tech. J., Vol. 61, No. 2 (February 1982), pp. 159-199.

9. __________ , "Volterra Expansions for Time-Varying Nonlinear Systems," Bell Syst. Tech. J. Vol. 61, No. 2 (February 1982), pp. 201-225.

10. L. V. Kantorovich and G. P. Akilov, *Functional Analysis in Normed Spaces,* New York: The Macmillan Company (1964).

GEOMETRIC STRUCTURE AND CONTROL DESIGN FOR A CLASS OF NONLINEAR SYSTEMS

D. K. LINDNER
Department of Electrical Engineering
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061

ABSTRACT

This paper considers the class of nonlinear systems of the form

$$\dot{x} = Ax + Bu + Hf(y)$$

$$y = Cx$$

Note that the nonlinearity is a function only of the output y rather than the total state x. Using chained aggregation, the geometric structure of this system is investigated. In particular, it is shown how linear partial state feedback can be used to decompose this system into a nonlinear system driving a linear system. This decomposes the control design into two steps, the first of which is nonlinear of reduced order. The second is a standard linear problem. An example is given.

1. INTRODUCTION

Design of controls for nonlinear systems remains a challenging problem. It is important that methods aimed at reducing the magnitude of the nonlinear synthesis problem should be considered. It is the purpose of this paper to propose a method for decomposition of the synthesis problem for a class of nonlinear control systems based on the system structure. By using feedback compensation the nonlinear synthesis problem is isolated and its magnitude is reduced.

The main idea is to extend the geometric structure identified by the Generalized Hessenberg Representation (GHR) [1-2] to a class of nonlinear systems characterized by linear dynamics and a collection of static nonlinearities. Because of the input-output nature of the GHR structure, we obtain a decomposition which makes explicit use of the nonlinearity's structure and it's interaction with the input. This leads directly to a decomposition of the system which isolates the nonlinearities in one subsystem. The importance of this decomposition is that it implies a natural decomposition of the control into three components: the first is a compensatory component that brings out the invariant structure and reduces the nonlinear system to a tandem configuration of an inherently nonlinear subsystem forcing a residual linear subsystem through static nonlinear interconnections; the second is a (possibly nonlinear) control componet that solves the synthesis problem associated with the inherently nonlinear part of the system; third is the residual control component that shapes the dynamics of the residual linear system. The decomposition represents an extension to nonlinear systems of the three term control decomposition scheme developed for linear systems and described in [2]. It is stressed that the major gain is in the decomposition of the nonlinear synthesis problem since nonlinear synthesis need be considered only in providing controls adequate for the inherently nonlinear part of the system.

This analysis also provides a partial invariant structure for this class of systems in that the linear subsystem contains a partial listing of the feedback invariants of the linear part of this nonlinear system [3-4]. The full implications of this have not yet been developed.

This paper is organized as follows. Section 2 reviews some basic geometric structure of the GHR. Section 3 extends this structure to a class of nonlinear systems. Section 4 describes a control design procedure. Section 5 suggests two extensions of the preceeding results. Section 6 contains an example. Section 7 has the conclusions.

2. PRELIMINARIES

In this section we briefly review the geometric structure of the GHR for the linear system

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{2.1}$$

where $x \varepsilon R^n$, $u \varepsilon R^m$ and $y \varepsilon R^r$. Define a linear transformation of the state

$$x = T\,\bar{x} \tag{2.2}$$

by

$$CT = [I \;\; 0] \tag{2.3}$$

Such a transformation exists if C is assumed full rank (as we do). Substituting (2.2) into (2.1) and partitioning compatibly with (2.3) we obtain

$$\begin{aligned} \dot{\bar{x}} = \begin{bmatrix} \dot{y} \\ \dot{x} \end{bmatrix} &= \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \bar{x} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u \\ y &= [I \;\; 0]\; \bar{x} \end{aligned} \tag{2.4}$$

This is the first step of chained aggregation which identifies the GHR [1,2]

We define the following well known [5] subspaces

$$L_i = \{ \xi = x(o) \mid y(o) = \dot{y}(o) = \cdots = y^{(i-1)}(0) = 0 \text{ when } u(t) \equiv 0 \}$$

We emphasize that these subspaces depend explicitly on C in (2.1). Furthermore, they are exhibited by the GHR. [2] In (2.4), for example, we have

$$\begin{aligned} L_1 &= N[C] = \text{sp} \begin{bmatrix} o \\ x_1 \end{bmatrix} \\ L_2 &= \{N[A_{12}] \hookrightarrow X \} \end{aligned} \tag{2.6}$$

To incorperate the input structure, suppose that B_1 is nonsingular. Then there exists a state feedback K such that

$$A + BK = \begin{bmatrix} A_{11} & 0 \\ A_{21} & \bar{A}_{22} \end{bmatrix} \tag{2.7}$$

Replacing (2.7) in the definition of L_i in (2.5) we get a new set of subspaces, which we denote L^K_i. These sets of subspaces reflect the input structure. In (2.7) we have

$$L^k_i = \text{sp} \begin{bmatrix} o \\ x_1 \end{bmatrix} \quad i = 1, 2, \cdots \tag{2.8}$$

It can be shown [1,2] that there is a choice of K which maximizes these sets of subspaces with respect to inclusion. Denote this set by L^*_{ℓ}. In this case, we have $L^*_n = V^*$, the supremal (A,B) - invariant subspace in N[C]. [6] In fact, this set is the one identified in (2.8) for the system (2.4) when B_1^{-1} exists.

For an arbitrary system, the subspaces L^*_i can be displayed explicitly using a modification of chained aggregation. It is shown in [2] that by using linear state and input transformations any system can be represented as

$$\begin{bmatrix} \dot{\bar{y}} \\ \dot{x}_r \end{bmatrix} = \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix} \begin{bmatrix} \bar{y} \\ x_r \end{bmatrix} + \begin{bmatrix} G_{11} & 0 \\ 0 & G_{22} \end{bmatrix} \begin{bmatrix} \bar{u} \\ \bar{u} \end{bmatrix} \qquad (2.9)$$

$$y = [C_1 \quad 0] \begin{bmatrix} \bar{y} \\ x_r \end{bmatrix}$$

where $\mathcal{R}[F_{12}] \subset \mathcal{N}[G_{11}]$ and $\mathcal{N}[G_{11}] = 0$. In [2] it is shown that

$$V^* = \mathrm{sp}\begin{bmatrix} 0 \\ x_r \end{bmatrix} \qquad (2.10)$$

and

$$\mathcal{R}^* = \langle F_{22} | G_{22} \rangle \qquad (2.11)$$

where $\mathcal{R}^*$ is the supremal controllability in N[C]. The representation in (2.9) suggests that we think of (2.1) as two interconnected subsystems: the aggregate subsystem which is given by

$$\dot{\bar{y}} = F_{11}\bar{y} + G_{11}\bar{u} + F_{12}x_r \qquad (2.12)$$

$$y = C_1\bar{y}$$

and the residual subsystem given by

$$\dot{x}_r = F_{22}x_r + G_{22}\tilde{u} + F_{21}\bar{y} \qquad (2.13)$$

The subspaces $\mathcal{V}^*$ and $\mathcal{R}^*$ describe a partial feedback invariant structure for (2.1) [3]. It consists of the controllability indices of (F_{22}, G_{22}) and the uncontrollable modes of that pair (the invariant zeros). Note that this structure is associated with the residual subsystem.

3. NONLINEAR SYSTEMS

For the rest of this paper we will consider nonlinear systems of the form

$$\dot{x} = Ax + Bu + D\,f(x) \qquad (3.1)$$

where the linear part of the system satisfies the assumptions for Section 2. The nonlinearities are contained in the function $f : R^r \rightarrow R^p$ such that

$f(o) = 0$. Furthermore, we assume that f is independent of several (most) of the state variables, i.e. $r < n$. We can identify this dependence by introducing an artifical output matrix in (3.1). The new system then becomes

$$\begin{aligned} \dot{x} &= Ax + Bu + D\,f(y), \\ y &= Cx. \end{aligned} \tag{3.2}$$

Thus, the matrix C defines the structure of the nonlinearity.

We can then extend the analysis of the last section directly to the nonlinear system (3.2). By defining the state and input space transformations with respect to the linear part of (3.2), this system can be represented as

$$\begin{bmatrix} \dot{\bar{y}} \\ \dot{x}_r \end{bmatrix} = \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix} \begin{bmatrix} \bar{y} \\ x_r \end{bmatrix} + \begin{bmatrix} G_{11} & 0 \\ 0 & G_{22} \end{bmatrix} \begin{bmatrix} \bar{u} \\ \tilde{u} \end{bmatrix} + \begin{bmatrix} D_1 \\ D_2 \end{bmatrix} f(y) \tag{3.3}$$

$$y = [C_1 \quad 0] \begin{bmatrix} \bar{y} \\ x_r \end{bmatrix}$$

Having incorperated the nonlinear structure in (3.2), we are able to use the geometric structure of the linear part of the system to decompose the nonlinear system. In (2.10) we identified the (A, B) - invariant subspace $\mathcal{V}^*$ for the linear part of (3.3). Using recent results for nonlinear systems [7], it is easy to show that

$$\operatorname{sp} \begin{bmatrix} 0 \\ x_r \end{bmatrix} \tag{3.4}$$

is an (A, B) invariant manifold for (3.3).

4. CONTROL SYNTHESIS

The analysis done in the last section leads to a synthesis procedure which for linear systems has been called Three Control Component Design (TCCD) [2]. This design procedure first decomposes the system into two subsystems such that the control design can be done sequentially on these two subsystems.

To apply this procedure to a nonlinear system, we first transform (3.2) into the representation (3.3). This guarantees the existence of a linear (partial) state feedback

$$\bar{u} = K_D x_r + \bar{v} \tag{4.1}$$

where K_D is chosen such that

$$F_{12}+G_{11}K_D = 0. \tag{4.2}$$

This yields a reduced order model of the output variables y given by the aggregate subsystem

$$\begin{aligned} \dot{\bar{y}} &= F_{11}\bar{y} + G_{11}\bar{v} + D_1 f(y) \\ y &= C_1\bar{y}. \end{aligned} \tag{4.3}$$

Note that the states x_r do not enter into (4.3) so that a control can be designed for the variables y using this reduced order model. The variables x_r are modeled by the residual subsystem

$$\dot{x}_r = F_{22}x_r+G_{22}u+F_{21}\bar{y}+D_2 f(y). \tag{4.4}$$

Once the (possibly nonlinear) control $\bar{v} = k(y)$ has been specified for the system (4.3), we can design a control for (4.4).

This synthesis procedure has several advantages. First, the control (4.1) is easily computed since it is linear. Second, the truly nonlinear aspect of the system (3.2) is contained in the aggregate subsystem. Thus, we have reduced the order of the nonlinear design and isolated its effect on the system. While a control synthesis for this subsystem is not, in general, easy, it has been simplified from the original problem. Also note that this component of the control can be linear or nonlinear as we like. Thirdly, the residual subsystem in (4.4) is a linear system. Thus the control of this system can be carried out using any one of a number of standard techniques.

As for linear systems, this procedure does not restrict the control design for the aggregate subsystems, but it imposes the invariant structure (discussed in Section 2) on the residual subsystem. This may impose an unacceptable design constraint; for instance, if one of the residuals fixed modes is unstable. However, by modifying the control (4.1) we can include the unstable modes in the aggregate system. This procedure is the same as the linear case. See [2] for details.

5. EXTENSIONS

The synthesis procedure described in Section 4 can be extended in several ways. The basic idea there is to decompose the linear part of the system based on the dependent variables of the nonlinearity. One specific decomposition was discussed but other decompositions are possible. Also, the structure of D, the input matrix for the nonlinearities, might be considered. For instance, the nonlinear part might be assigned to the residual subsystem or the two subsystems might be decoupled.

The second extension occurs when the full state is not available for feedback. Suppose that the outputs y in (3.2) are really measured variables. Then it turns out we can easily construct a linear observer for this system. By a state transformation, the system (3.2) can be put in the form

$$\begin{bmatrix} y \\ \dot{x}_r \end{bmatrix} = \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix} \begin{bmatrix} y \\ x_r \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u + \begin{bmatrix} D_1 \\ D_2 \end{bmatrix} f(y)$$

$$y = [I \ 0] \begin{bmatrix} y \\ x_r \end{bmatrix} . \qquad (5.1)$$

A reduced order observer for the variables x_r is given by

$$\dot{\hat{x}}_r = F_{22}\hat{x}_r + F_{21}y + D_2 f(y) + B_2 u. \qquad (5.2)$$

The dynamics of the observer are governed by F_{22} and these can be arbitrarily chosen by first transforming the state of (5.2) by

$$\begin{bmatrix} y \\ x_r \end{bmatrix} = \begin{bmatrix} I & 0 \\ X & I \end{bmatrix} \begin{bmatrix} y \\ \bar{x}_r \end{bmatrix} . \qquad (5.3)$$

This is a particularly simple observer structure for a nonlinear system.

6. EXAMPLE

In this section we will illustrate the ideas above by considering the aircraft landing problem described in [8].

The problem is to design an automatic landing control for a heavy transport aircraft during the final landing stage, the flareout phase. Typical state equations are

$$\dot{x}_1 = 0.41x_1 + 0.381x_2 - 0.562x_3 - 2.552\,x_7 + 0.221f(x_5)$$

$$\dot{x}_2 = -0.066x_1 - 0.577x_2 + x_3 - 0.05x_6 - .992x_7 - 0.395f\,(x_5)$$

$$\dot{x}_3 = 0.011x_1 - 1.108x_2 - 01822x_3 - 1.264x_6 - 0.157x_7 - 3.544f(x_5)$$

$$\dot{x}_4 = x_3$$

$$\dot{x}_5 = -12.147 + 4.049\,(x_4 - x_2)$$

$$\dot{x}_6 = u_1$$

$$\dot{x}_7 = (u_2 - x_7)\,/\,2$$

where the seven elements of the state vector x are as follows.

x_1 = increment of forward speed (ft/sec)

x_2 = increment of altitude angle (deg)

x_3 = increment of pitch rate (deg/sec)

x_4 = increment of pitch angle (deg)

x_5 = height (ft)

x_6 = increment of elevator angle (deg)

x_7 = increment of throttle

u_1 = increment of elevator rate (deg/sec)

u_2 = input to throttle actuator

and f, the nonlinear ground effect term, is

$$f = 400/(3x_5 + 100) - 1. \qquad (6.2)$$

We want to select controls u_1 and u_2 such that x_1 (the forward speed) and x_5 (the height) follow a prescribed trajectory while x_4 (the pitch angle) should have a final value which is positive. We shall take the simple point of view that x_1, x_4, and x_5 are the variables of primary interest. A complete description of this problem can be found in [8].

We shall use the TCCD to decompose this problem into its linear and nonlinear subsystems. Equation (6.1) shows that the nonlinearity depends on the single state variable x_5. We formally identify this structure by introducing an output equation

$$y = [0\ 0\ 0\ 0\ 1\ 0\ 0]x. \qquad (6.3$$

Combine (6.3) with (6.1) and then use the transformation procedure described in [2] into the representation (3.2). The resulting decomposition is

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \\ \dot{y}_3 \end{bmatrix} = \begin{bmatrix} 0 & -4.049 & 0 \\ 0 & -0.5777 & 1 \\ 0 & -0.025 & -0.6665 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -0.0413 & -0.4102 & -0.5406 & -0.0333 \end{bmatrix} \begin{bmatrix} x_1 \\ x_4 \\ x_3 \\ x_6 \end{bmatrix}$$

$$+ \begin{bmatrix} 0 \\ 0 \\ -.5 \end{bmatrix} u_2 + \begin{bmatrix} 0 \\ -0.395 \\ -0.0179 \end{bmatrix} f(y_1) + \begin{bmatrix} -12.147 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_4 \\ \dot{x}_3 \\ \dot{x}_6 \end{bmatrix} = \begin{bmatrix} -0.1255 & 1.8380 & -0.562 & 0.1261 \\ 0 & 0 & 1 & 0 \\ 0.0214 & -1.0173 & -0.822 & -1.256 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_4 \\ x_3 \\ x_6 \end{bmatrix} \qquad (6.4)$$

$$+ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u_1 + \begin{bmatrix} 0 & 0.381 & 2.522 \\ 0 & 0 & 0 \\ 0 & -1.108 & 0.157 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

$$+ \begin{bmatrix} 0.271 \\ 0 \\ -3.544 \\ 0 \end{bmatrix} f(y)$$

with

$$y_1 = x_5$$
$$y_2 = x_2 - x_4$$
$$y_3 = -x_7 - 0.066x_1 - 0.5777x_4 - -0.05x_6.$$
$$\hat{u}_2 = .1u_1 + u_2$$

The partitioning in (6.4) corresponds to the partitioning in (3.2).

It is interesting to note that the controls seem to decompose naturally with respect to the state decomposition in (6.4) in the sense that u_2 affects only the aggregate subsystem and the first control's effect on the aggregate is an order of magnitude less than u_2.

With this in mind the first step is to compute a decoupling control $\hat{u}_2 = K_D x_r + \hat{v}_2$

$$K_D = [-.0826 \quad -.8204 \quad -1.0812 \quad -.0666]. \tag{6.6}$$

When this control is applied to (6.4), we obtain the aggregate subsystem

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \\ \dot{y}_3 \end{bmatrix} = \begin{bmatrix} 0 & -4.049 & 0 \\ 0 & -0.5777 & 1 \\ 0 & -0.025 & -0.6665 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -0.5 \end{bmatrix} v_2 + \begin{bmatrix} 0 \\ -0.395 \\ -0.0179 \end{bmatrix} f(y_1) + \begin{bmatrix} -12.147 \\ 0 \\ 0 \end{bmatrix}. \tag{6.7}$$

This subsystem, which represents the nonlinear part of the model, forms a reduced order model for the height variable y_1 $(=x_5)$. Thus we can design a control $v_2 = L(y)$ so that the aircraft follows in the prescribed height trajectory.

Following computation of the aggregate control, we can design the residual system control using the model

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_4 \\ \dot{x}_3 \\ \dot{x}_6 \end{bmatrix} = \begin{bmatrix} -0.1255 & 1.8380 & -0.562 & 0.1261 \\ 0 & 0 & 1 & 0 \\ 0.0214 & -1.0173 & -0.822 & -1.256 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_4 \\ x_3 \\ x_6 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u_1 + \begin{bmatrix} 0.271 \\ 0 \\ -3.544 \\ 0 \end{bmatrix} f(y_1) + \begin{bmatrix} 0 & 0.381 & 2.522 \\ 0 & 0 & 0 \\ 0 & -1.108 & 0.157 \\ 0 & 0 & 0 \end{bmatrix} y. \tag{6.8}$$

Since this system is controllable, we can select the control to meet the specifications for forward speed (x_1) and pitch angle (x_4).

The residual subsystem (6.8) also contains the partial invariant structure described in Section 2. In this case, the linear part of (6.8) is a single input controllable pair. Thus, the invariant is the dimension of the state space, i.e. 4. Here, the invariant structure imposes no design limitations.

7. CONCLUSIONS

In this paper we have discussed a synthesis procedure for a class of nonlinear systems. The major idea is to use the invariant structure of the system to decompose the system into two smaller problems: one linear and one nonlinear. Thus, the magnitude of the (difficult) nonlinear synthesis is reduced.

The direction of this work is to extend known linear system properties to nonlinear systems. This provides a framework in which to analyze the effect of nonlinearities or widely used linear procedures. Thus, this framework should be useful for studying other problems as well.

Acknowledgment

The author acknowledges several useful discussions with Prof. J. Medanic and Prof. W. R. Perkins that contributed to the development of this paper.

8. REFERENCES

1. D. K. Lindner, "Chained Aggregation and Control System Design: A Geometric Approach", Report R-966 (DC-56) Decision and Control Laboratory, University of Illinois, Urbana, Illinois, 1982. Ph.D. thesis.

2. D. K. Lindner, W. R. Perkins and J. Medanic, "Chained Aggregation and Three-Control Component Design: A Geometric Analysis", Int. J. Control, Vol. 35, 1982, pp. 621-639.

3. A. S. Morse, Structural Invariants of Linear Multivariable Systems" SIAM J. Control, Vol. 11, 1973, pp. 446-465.

4. D. K. Linder, J. Medanic and W. R. Perkins, "An Algorithm for Computing the Structural Invariants of Linear Systems", Proc. Systems Engineering for Power: Organization Forms for Large Scale Systems, Vol II, Davis, Switzerland, 1979, pp. 3.59-3. 72.

5. B. P. Molinari, "A Strong Controllability and Observability in Linear Multivariable Control," IEEE Trans, on Automatic Control, Vol. AC-21, 1976, pp. 761-764.

6. W. M. Wonham, Linear Multivariable Control, Springer, Berlin, 1979.

7. A. Isidori, A. Krener, C. Gori-Giorgi, and S. Monaco, "Nonlinear Decoupling via Feedback: A Differential Geometric Approach" IEEE Trans on Automatic Control, Vol. AC - 26, 1981, pp. 331-345.

8. Dyer and McReynolds, The Computation and Theory of Optimal Control, Academic Press, New York, 1970.

OBSERVERS FOR NONLINEAR/TIME-VARYING SYSTEMS

E. Emre
Department of Electrical Engineering/Computer Science
Texas Tech University
Lubbock, Texas 79409

1. INTRODUCTION

Design of observers (asymptotic state estimators) and their relation to properties of systems such as observability is one of the well known fundamental problems of system-control theory. Both observability and construction of observers are well studied subjects for linear time-invariant systems (see [1] for an explicit global parametrization of all observers for linear time-invariant systems).

However, for nonlinear and/or time-varying systems the situation is very different. Although the problem of observability (several types) has been studied in some detail for nonlinear systems, this is mostly in relation to realization theory (see e.g., [2], [4], [5]). Existence and construction of observers for nonlinear systems and the relation of this problem to the observability conditions in the literature has not been clear thus far. The main problem here has been the lack of special structures in nonlinear systems, and the previous authors considering these problems for nonlinear systems have been unable to determine a suitable structure for observers that can be used for a large class of nonlinear systems.

Previous approaches to construction of observers have been mainly concerned with bilinear systems, linearizable systems or systems for which results to obtain a nonlinear filter are available (see e.g., [8], [3], and [7] respectively). All of these are clearly restricted classes of nonlinear systems, and in the literature there exist no previous results on a reasonably general structure for observers for nonlinear systems.

The purpose of this paper is to introduce a structure for observers which is applicable to a nontrivially large class of nonlinear time-varying systems, (both discrete-time and continuous-time), and to study the existence conditions also relating them to some of the observability conditions studied in the literature. Our observers generalize the idea of polynomial fractional descriptions well known for linear systems (see e.g., [1]), to nonlinear systems. With our formulation, the observer problem is considerably simplified and clarified.

In section 2, we define our observer structure which is a valid structure for general nonlinear time-varying systems. Under certain conditions, these observers can be chosen to be quasilinear (in the sense that the dynamic part of the observer is linear), thereby simplifying the stability issues involved, but (as we show) they still exist for a considerably large class of nonlinear systems. For us, an observer is a causal dynamic system which asymptotically estimates the state of a given system based on its inputs and outputs. Our observers have the property that (when they exist) the error between the state of the system and its estimate obeys a stable linear differential (difference) equation. Despite having this very desirable property, (as we show) such observers work (estimate the state) for nonlinear time-varying systems under those inputs which distinguish every pair of states of the original system. In fact, in

---

This research was supported in part by the U.S. Air Force Office of Scientific Research Grant AFOSR-82-0282 and in part by the National Science Foundation under NSF Research Grant ECS-8204512.

this case state estimation can be done with arbitrary rate of (exponential) convergence. In particular, if the given system is observable for every input (i.e., if the observability map is left invertible for every input), such observers (with arbitrary rate of convergence) always exist. The conditions for this kind of observability have been studied in the literature in detail previously. For example, in the case of systems over infinitely differentiable functions, one can use the "ratio condition" on the Jacobian of the observability map (see [5,6,9] and the references there for details). Also, clearly, if a nonlinear system can be exactly transformed into an observable linear system, then it has to satisfy our existence conditions. Thus, with our techniques, one can design an observer of the type introduced here for such a system without necessarily transforming the system into a linear system, and linearization is not necessarily needed. Hence, our observers exist for a much larger class of observable nonlinear systems than studied previously. Furthermore, under the existence conditions, we give a characterization of all possible such observers. This provides flexibility for choice of particular types of observers which may be more suitable for a particular purpose. Thus one can conceive of an optimum choice of the observer thereby obtaining a nonlinear filter (as it is well known, a recursive filter for a linear system can be viewed as an optimal observer).

Furthermore, one can also obtain detectability conditions for the existence of such observers, as well as conditions for unknown input detectability and observability. Straightforward specialization of our general approach established here to linear time-varying systems and linear systems over noncommutative rings leads to detectability, unknown input detectability and observability conditions which did not appear prior to this paper. This provides solutions to these most of which were previously unsolved problems, thereby showing the power of our approach very clearly. Also, our results are applicable to both continuous-time and discrete-time systems, and thus provide a unifying solution to these problems. In the case of linear systems over commutative rings, this provides a time-domain interpretation of the detectability results previously obtained using a polynomial fractional approach. For more details of these, we refer to Emre [10].

2. OBSERVERS FOR NONLINEAR/TIME-VARYING SYSTEMS

We will first consider nonlinear discrete-time systems, and comment on the necessary modifications to apply our approach to continuous-time systems. Consider the nonlinear discrete-time time-varying system S:

$$x(t+1) = f(x(t),u(t),t)$$

$$y(t) = h(x(t),u(t),t) \quad ; t=0,1,\ldots$$

where $x(t) \varepsilon R^n$, $y(t) \varepsilon R^p$, $u(t) \varepsilon R^m$. Next we will define a quasilinear observer, S_0, for such a system S.

2.1. DEFINITION. A quasilinear observer, S_0, for S is a dynamical system described as

(2.2.) $L(t,p)\hat{x}(t) = M(t;u(t),pu(t),\ldots,p^r u(t);y(t),py(t),\ldots,p^r y(t))$

(where p is the shift operator; $pg(t) := g(t+1)$ for any function g, and

(2.3) $L(t,p) := I_n p^r + \sum_{j=0}^{r-1} A_j(t) p^j$,)

such that

(i) for any integer $t_0 \geq 0$, $\hat{x}(t) = x(t)$ for $t = t_0, t_0+1, \ldots, t_0 + r-1$, implies that $\hat{x}(t) = x(t)$ for all $t \geq 0$, for every input u, and such that (ii) for any integer $t_0 \geq 0$, no matter what values $\hat{x}(t)$ and $x(t)$ may have for $t = t_0, t_0 + 1, \ldots, t_0 + r-1$,

$$\lim_{t\to\infty}(\hat{x}(t)-x(t)) = 0$$

for every input u.

2.4. DEFINITION. An operator $L(t,p)$ as in (2.3) is called stable iff the corresponding autonomous system

$$L(t,p)\hat{x}(t) = 0$$

is exponentially stable (i.e., for every $t_0 \geq 0$ and for every values of $\hat{x}(t)$ for $t = t_0, t_0 + 1, \ldots, t_0 + r-1, \hat{x}(t)\to 0$ exponentially). Next we define detectability of S in a natural way.

2.5. DEFINITION. S is called (quasilinear) detectable iff there exists an observer as above.

The next theorem yields a necessary and sufficient condition for S to be detectable. For this, first write

$$(2.6) \qquad X(t) := \begin{bmatrix} x(t) \\ px(t) \\ \vdots \\ p^r x(t) \end{bmatrix} = F_r(x(t);u(t),pu(t),\ldots,p^r u(t);t),$$

$$(2.7) \qquad Y(t) := \begin{bmatrix} y(t) \\ py(t) \\ \vdots \\ p^r y(t) \end{bmatrix} = H_r(x(t);u(t),pu(t),\ldots,p^r u(t);t),$$

and

$$U(t) := \begin{bmatrix} u(t) \\ pu(t) \\ \vdots \\ p^r u(t) \end{bmatrix}$$

Clearly, the functions F_r and H_r can be obtained from S by successive application of the operator p on x(t) and y(t).

Now we state our necessary and sufficient conditions for the existence of observers.

2.8. THEOREM. S is (quasilinear) detectable iff for some integer r there exist nXn matrices $A_i(t)$; $i=0,1,\ldots,r-1$, and a function K such that

$$(2.9) \qquad [A_0(t):\ldots:A_{r-1}(t):I_n]F_r(x(t);U(t);t)$$
$$= K(t;U(t);H_r(x(t);U(t);t))$$

for all $t \geq 0$, and for all u, and such that the operator

$$L(t,p) := \sum_{j=0}^{r-1} A_j(t)p^j + I_n p^r$$

is a stable operator. (Note that K is required to be defined on only a subset of $Z_+ \times R^{(r+1)m} \times R^{(r+1)p}$).

PROOF. Let S_0 be an observer as in (2.2) for S. Suppose that for arbitrary fixed t_0, for $t=t_0,t_0+1,\ldots,t_0+r-1$, we have $\hat{x}(t)=x(t)$. Then we have $\hat{x}(t)=x(t)$ for all $t \geq t_0$, and for all inputs u. Thus

$$(2.9a)\qquad L(t,p)x(t) = M(t;U(t);Y(t))$$

for all $t \geq t_0$ and every input u. Now substituting (2.6,7) into (2.9a), and letting $t_0 = 0$, we obtain (2.9) with K defined as

$$K(t;U(t);H_r(t;x(t);U(t))):=M(t;U(t);H_r(t;x(t);U(t)))$$

for all $t \geq 0$, and every input u. On the other hand, if $\hat{x}(t)$ is a solution of (2.2) for $t \geq t_0 \geq 0$ with arbitrary initial conditions $\hat{x}(t_0),\ldots,\hat{x}(t_0+r-1)$, then

$$L(t,p)(\hat{x}(t)-x(t)) = 0$$

for $t \geq 0$, where t_0 can be arbitrary. Since S_0 is an observer for S, $(\hat{x}(t)-x(t))\to 0$ exponentially. As the initial conditions of $\hat{x}$ as well as t_0 are arbitrary, $L(t,p)$ has to be a stable operator.

Conversely, suppose that (2.9) is satisfied for $t \geq 0$, and that $L(t,p)$ is stable. Let $t_0 \geq 0$ be an integer. (2.9) implies that $\hat{x}(t) = x(t)$ is a solution of

$$(2.10)\qquad L(t,p)\hat{x}(t) = K(t;U(t);H_r(t;x(t),U(t)))$$

for $t \geq t_0$ satisfying the initial conditions $\hat{x}(t) = x(t)$ for $t = t_0,t_0+1,\ldots,t_0+r-1$, for any u. Since $L(t,p)$ is a monic operator, (2.10) has a unique solution for each set of initial conditions and for each given u. Thus, if for given u, $\hat{x}_1(t)$ is a solution corresponding to a different set of initial conditions, we have

$$L(t,p)(\hat{x}_1(t)-x(t)) = 0$$

for $t \geq t_0$. As $L(t,p)$ is stable, $\hat{x}_1(t)-x(t)\to 0$ exponentially. Hence the proof. □

Although detectability is a necessary and sufficient condition for quasilinear observers to exist, it is also important to be able to guarantee the existence of observers with desired rate of convergence. This motivates the following definition of observability.

2.11. DEFINITION. S is called observable iff for some integer $r\geq 0$, for every operator $L(t,p)$ of degree r as in Theorem 2.8, there exists a corresponding K as in (2.9) such that (2.9) is satisfied.

The following corollary follows from Theorem 2.8 and the above definition.

2.12. COROLLARY. S is observable iff for some integer $r\geq 0$ and for each $i=0,1,\ldots,r$, there exists a function K_i such that

$$(\hat{p}_i \cdot F_r)(x(t);U(t);t) = K_i(t;U(t);H_r(x(t);U(t);t))$$

for all $t\geq 0$, and for every input u. (Here $\hat{p}_i$ is the projection defined by

$$(\hat{p}_i \cdot F_r)(x(t);U(t);t) = p^i x(t)$$

for $i=0,1,\ldots,r$.)

2.13. REMARK. The functions $\hat{p}_i \cdot F_r$ occurring above can be obtained successively from S as F_r and H_r are obtained.

2.14. REMARK. For nonlinear systems detectability and observability are dependent on the trajectories of the system S, and thus it may be difficult to check these conditions. This motivates the following definitions of strong detectability and strong observability, which are independent of this.

2.15. DEFINITION. S is strongly detectable iff for some $r\geq 0$, the condition of Definition 2.11 is satisfied with $x,u,\ldots,p^r u,t$ viewed as independent variables.

It is clear that strong detectability (observability) implies detectability (observability). The following theorem connects our definition of observability (i.e., being able to determine the initial state uniquely from the knowledge of y(t) and u(t) over a finite time interval).

2.16. THEOREM. Suppose that for some integer r, there exists a function

$$H_l(Y(t);U(t);t)$$

such that

$$H_l(H_r(x(t);U(t);t);U(t);t) = x(t)$$

for all $t \geq 0$ and for every input u. Then the system S is observable.

PROOF. Clearly, if we let

$$K_i(t;U(t);Y(t)) = (\hat{p}^i \cdot F_r)(H_l(Y(t);U(t);t);U(t);t),$$

then the condition in (2.12) will be satisfied. Thus the system S is observable. □

2.17. REMARK. Clearly, if the above condition is satisfied with $x,u,\ldots,p^r u,t$ viewed as independent variables, the system S is strongly observable. Also note that if S is either time-invariant or with no inputs, again all our results are valid by omitting t and/or $p^i u$'s appearing thus far.

2.18. REMARK. It also follows that if S is observable or strongly observable, then there exists an observer with arbitrary dynamics (i.e. arbitrary L(t,p)). In fact L(t,p) can be chosen to be $a(p) \cdot I_n$, where a(p) is any monic polynomial with constant real coefficients of degree r with roots inside the unit circle of the complex plane. It is also clear that observability is a necessary and sufficient condition for the condition of Theorem 2.16.

2.19. REMARK. Our observability conditions are clearly left invertibility conditions on the observability map of the system S. For conditions for left invertibility of functions, we refer to [5,6,9] and the references given there.

2.20. REMARK. It is clear that our definitions and results in this section can be modified to be applicable to continuous time systems (e.g., let p denote the derivative operator instead of the shift operator, and let t take on real values, put conditions of uniqueness, differentiability etc.).

2.21. REMARK. We have considered quasilinear observers of the form

$$L(t,p)x = M(U,Y,t)$$

for simplicity. However, our approach is clearly also suitable for considering observers where L can be a non-linear difference or differential operator satisfying certain properties so that the resulting system is an asymptotic state estimator.

Also one can restrict the inputs to a class of "admissible inputs" to obtain observers which work only for these particular inputs. Clearly in this case our theorems can be modified simply putting the phrase: "for all admissible inputs" instead of"for all u." One can do further modifications by requiring the functions K and/or K_i's to be of a certain class. For example if the nonlinearity is polynomial type, one can require the functions K and/or K_i to be polynomials in $u,pu,\ldots,p^r u$, $y,py,\ldots,p^r y$, etc. These can be based on the approach we have established here.

2.22. REMARK. From our approach one can also obtain unknown input detectability (obsrvability) conditions rather easily. These are conditions for a quasilinear observer to exist which only uses the outputs of the plant (but not the inputs) to a symptotically estimate the state of the given system.

2.23. THEOREM. There exists a system described as $L(t,p)\hat{x}(t) = M(t,Y)t))$ where L(t,p) is linear such that

i) $\lim(\hat{x}(t)-x(t)) = 0$, exponentially for all initial values of $\hat{x}$ and x, and such that

ii) $\hat{x}(t) = x(t)$

whenever the initial values match (i.e., S is unknown input detectable) iff for some integer r there exist nxn matrices $A_i(t);i = 0,1,\ldots r-1$, and a function K such that

$$[A_0(t):\dots:A_{r-1}(t):I_n]F_r(x(t);U(t),t)$$
$$= K(t;Hr(x(t),U(t);t))$$

for all $t \geq 0$ *and for all* u, *and such that the operator*

$$L(t,p) := \sum_{j=0}^{r-1} Aj(t)p^j + I_n p^r$$

isa stable operator.

An observer of the above type exists such that $\hat{x}(t)-x(t)$ *can be made to approach zero exponentially with arbitrary rate of convergence (i.e.,* S *is unknown input observable) iff for some integer* r *there exists a function*

$$H_1(Y(t);t)$$

such that

$$H_1(H_r(x(t);U(t);t);t) = F_r(x(t), U(t),t)$$

for all $t \geq 0$ *and for every input* u.

3. LINEAR TIME-VARYING SYSTEMS

In this section the general approach developed in Section 2 is specialized to linear time-varying systems to obtain much simpler conditions for the existence and construction of observers. We should note that the results of this section are also applicable to linear systems over (not necessarily commutative) rings, and from these results one can obtain the algebraic detectability criteria known for linear systems over rings.

Consider a linear time-varying system S

$$x(t+1) = F(t)x(t) + G(t)u(t)$$

(3.1)

$$y(t) = H(t)x(t) + J(t)u(t).$$

In this case the expressions for F_r and H_r are considerably simpler, and are of the form

$$\begin{bmatrix} x(t)\\ px(t)\\ \vdots\\ p^r x(t)\end{bmatrix} = \begin{bmatrix} I & 0 & \cdots & \cdots & 0\\ F_{11}(t) & F_{12}(t) & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ F_{r1}(t) & F_{r2}(t) & \cdots & \cdots & F_{rr+1}(t)\end{bmatrix}\begin{bmatrix}x(t)\\u(t)\\ \vdots \\ p^{r-1}u(t)\end{bmatrix},$$

(3.2)

$$\begin{bmatrix} y(t)\\ py(t)\\ \vdots\\ p^r y(t)\end{bmatrix} = \begin{bmatrix} H_{01}(t) & H_{02}(t) & 0 & \cdots & \cdots & 0\\ H_{11}(t) & H_{12}(t) & H_{13}(t) & 0 & \cdots & 0 \\ \vdots & \vdots & & \ddots & 0 & \\ \vdots & \vdots & & & \ddots & \\ H_{r1}(t) & H_{r2}(t) & \cdots & \cdots & \cdots & H_{rr+2}(t)\end{bmatrix}\begin{bmatrix}x(t)\\u(t)\\ \vdots \\ p^{r}u(t)\end{bmatrix},$$

where the functions F_{ij}'s and H_{ij}'s can be obtained by successive application of the shift operator p on (3.1).

Then Theorem 2.8 in Section 2 specializes the following theroem.

3.3. THEOREM. *There exists a linear time-varying observer of the form*

$$(3.4)\quad (I_n p^r + \sum_{i=0}^{r-1} A_i(t)p^i)\hat{x}(t) = (\sum_{i=0}^{r} B_i(t)p^i(y(t) + (\sum_{i=0}^{r} C_i\ (t)p^i)u(t)$$

(where x̂t) is an asymptotic estimate of x(t) for all initial conditions) for S in (3.1.1) iff there exists a time-varying matrix $T_1(t)$ (of suitable dimensions) such that

$$(3.5)\quad [A_0(t):\ldots:A_{r-1}(t):I_n]\begin{bmatrix} I_n \\ F_{11}(t) \\ F_{12}(t) \\ \vdots \\ F_{r1}(t)\end{bmatrix} = T_1(t)\begin{bmatrix} H_{01}(t) \\ H_{11}(t) \\ \vdots \\ H_{r1}(t)\end{bmatrix}$$

for all $t \geq 0$, where $L(t,p) := \sum_{i=0}^{r-1} A_i(t)p^i + I_np^r$ is a stable operator.

PROOF. Using the same approach for the proof of Theorem 2.8, using additivity of the terms involving x(t) and u(t). □

3.6. REMARK. Clearly the same technique can be used for continuous-time time-varying systems, replacing the shift operator by the derivative operator as remarked in the previous section.

3.7. REMARK. It is clear that if the observability matrix

$$\begin{bmatrix} H_{01}(t) \\ H_{11}(t) \\ \vdots \\ H_{r1}(t)\end{bmatrix}$$

is left invrtible (i.e. it has a time varying left inverse) then L(t,p) can be chosen arbitrarily. In particular it can be chosen to be L(t,p) = a(p) where a(p) is any polynomial with constant real coefficients of degree r with roots in the unit circle.

3.8. REMARK. Clearly we can easily specialize Theorem 2.23 to linear time-varying systems, again using additivity, to obtain criteria for unknown input detectability and observability for linear time-varying systems.

3.9. THEOREM. There exists a linear time-varying observer

$$L(t,p)\hat{x}(t) = M(t)y(t)$$

such that

i) $\hat{x}(t)-x(t)\to 0$ exponentially for all initial values of $\hat{x}$ and x, and such that

ii) $\hat{x}(t) = x(t)$ whenever the initial values match (i.e., S is unknown input detectable) iff, for some integer r, there exist nxn matrices $A_i(t)$ and a matrix $T_1(t)$ (of suitable dimensions) $i = 0,1,\ldots,r-1$, such that

$$(3.10a)\quad [A_0(t):\ldots:A_{r-1}(t):In]\begin{bmatrix} I_n \\ F_{11}(t) \\ \vdots \\ F_{r1}(t)\end{bmatrix} = T_1(t)\begin{bmatrix} H_{01}(t) \\ \vdots \\ H_{r1}(t)\end{bmatrix}$$

and

$$(3.10b)\quad [A_0(t):\dots:A_{r-1}(t):I]\begin{bmatrix} 0 & . & 0 & . & . & . & . & . & . & .0 \\ F_{11}(t) & F_{12}(t) & 0 & . & . & . & & & & .0 \\ \vdots & & & & & & & & & \\ F_{r1}(t) & F_{r2}(t) & \dots & & & & & & & F_{rr+1}(t) \end{bmatrix}$$

$$= T_1(t)\begin{bmatrix} H_{02}(t) & 0 & . & . & . & . & 0 & \\ \vdots & & & & & & & 0 \\ H_{r-1,2}(t) & & . & & & H_{r-1,r+1}(t) & & \\ H_{r2}(t) & . & . & . & . & . & H_{r,r+1} & \end{bmatrix}$$

$$T_1(t)\begin{bmatrix} 0 \\ \vdots \\ 0 \\ H_{rr+2}(t) \end{bmatrix} = 0$$

for all $t \geq 0$, where $L(t,p) := \sum_{i=0}^{r-1} A_i(t)p^i + I_n p^r$ is a stable operator.

An observer of the above type exists such that $\hat{x}(t)-x(t)$ can be made to approach zero exponentially with arbitrary rate of convergence (i.e., S is unknown input observable) iff there exists a matrix L(t) such that

$$(3.11)\quad \begin{bmatrix} I & 0 & 0 & . & . & . & . & . & . & .0 \\ F_{11}(t) & F_{12}(t) & 0 & . & . & . & . & . & . & .0 \\ \vdots & \vdots & & & & & & & & \\ F_{r1}(t) & F_{r2}(t) & . & . & . & . & . & & & F_{rr+1}(t) \end{bmatrix} = L(t)\begin{bmatrix} H_{01}(t) & H_{02}(t) & 0 & \dots & 0 \\ \vdots & \vdots & & & . \\ H_{r1}(t) & H_{r2}(t) & \dots & . & H_{rr+1}(t) \end{bmatrix}$$

$$L(t)\begin{bmatrix} 0 \\ \vdots \\ 0 \\ H_{rr+2}(t) \end{bmatrix} = 0\ .$$

4. REFERENCES

[1] E. Emre, "Explicit (global) parametrization of all stabilizing compensators and observers for linear feedback systems," Proc. IFAC Symposium on Comp. Aid. Des. of Multivariable Technological Systems, Purdue Univ., Sept. 1982.

[2] P. D'Alessandro, A. Isidori and A. Ruberti, "Realization and structure theory of bilinear dynamical systems," SIAM J. Contr., vol. 12, No. 3, Aug. 1974.

[3] A. J. Krener and A. Isidori, "Linearization by output injection and nonlinear observers," Preprint, Dept. of Math., Univ. of California, Davis, 1983.

[4] R. Hermann and A. Krener, "Nonlinear controllability and observability," IEEE Trans. Autom. Contr., 22, 5, Oct. 1977.

[5] S. R. Kou, D. L. Elliott and T. J. Tarn, "Observability of nonlinear systems," Inf. and Contr., 22, 89-98, 1973.

[6] J. P. Gauthier and G. Bornard, "Observability for any u(t) of a class of nonlinear systems," Proc. IEEE Conf. on Decis. and Contr., pp. 910-915, 1980.

[7] J. S. Baras and P. S. Krishnaprasad, "Dynamic observers as asymptotic limits of recursive filters," IEEE Conf. Decis. Contr., 1982.

[8] Hara and Furuta, "Minimal order state-observers for bilinear systems," Int. J. Contr., Vol. 24, 1976.

[9] W. M. Boothby, An Introduction to Differentiable Manifolds and Reimanian Geometry, Academic Press, New York, 1975.

[10] E. Emre, "Observers for nonlinear time-varying systems and linear systems over noncommutative rings," Nov. 1982. Preprint. Dept. of Elec. Engr./Comp. Sci., Texas Tech University.

ON NULL CONTROLLABILITY OF NONLINEAR SYSTEMS ARISING IN AMCD-SPACECRAFT ATTITUDE CONTROL SYSTEM

A. S. C. SINHA
Purdue University at Indianapolis
Indianapolis, Indiana 46223

ABSTRACT

The results determine the region of null controllability for a class of nonlinear control problems. Sufficient conditions on nonlinearities are such that the domain C of null controllability has been obtained. The result has an application in AMCD-spacecraft attitude control system.

1. INTRODUCTION

Consider the nonlinear perturbed control system

$$\frac{dx}{dt} = f(x,u,t) + g(x,u,t) \tag{1.1}$$

where $f(x,u,t)$, $g(x,u,t) \in C^1$ (R^{n+m}) for each $t \in I_0$ with $I_0 = (-\infty,\infty)$. It is assumed throughout the sequel that u belongs to the set of all bounded measurable control functions with values $u(t)$ in Ω, a subset of R^m. Such controls will be called admissible. The result has application in AMCD-spacecraft attitude control system (Sinha 1978, Anderson et.al., 1979, 1975).

Define the domains C of null controllability of (1.1) to be the set of all initial points $x_0 \in R^n$ such that if $u(t)$ is an admissible control and $x(t)$ a solution of

$$\frac{dx}{dt} = f(x(t),u(t),t) + g(x(t),u(t),t) \text{ with } x(t_0) = x_0 \tag{1.2}$$

then at some finite time $t_1 \geq t_0$ we have $x(t_1) = 0$. If C contains an open neighborhood of zero then (1.1) is said to be locally controllable to the origin.

The problem of interest here is as follows: determine sufficient conditions on f and g such that the domain C of null controllability for (1.1) is the whole of R^n. If, for example, x denotes the error in some physical system described by the equation (1.1), a problem of considerable practical importance is that of steering, in finite time, all initial states to one fixed state, the origin.

For the autonomous system in R^n of type

$$\frac{dx}{dt} = Ax + Bu + h(x,u) \tag{1.3}$$

where h together with its partial derivatives are continuous in R^{n+m} and zero at the origin in R^{n+m}, and where the admissible controls are assumed continuous, Lukes (1972) proved the following Theroem:

Theorem (Lukes). If for the control system (1.3) in R^n

(i) rank $[B, \; AB, \; . \; . \; . \; . \; , \; A^{n-1} \; B] = n$;

(ii) h together with its partial derivatives are continuous in R^{n+m} and zero at the origin in R^{n+m};

(iii) $||h(x,u)||$ is bounded in R^{n+m}.

Then the domain of null controllability of (1.3) is R^n.

Remark

It must be pointed out that the class of admissible controls of Lukes can be unbounded. As a result he does not require explicitly that A be a stability matrix. In our case where the controls are bounded, an analogous statement of Lukes Theorem will require that A be stable.

The present paper proves the same conclusion for the system (1.1) with bounded controls. The required result, Theorem 3 is an immediate consequence of Corollary 1 and Theorem 2: every solution of (1.1) arrives in a neighborhood of zero in some finite time T, by Corollary 1; and is steered finally to the origin at time $t_1 + T$, because (1.1) is locally controllable. The verification of Corollary 1 is an adaptation of Ezeilo (1966) and Chukwu (1974), while Theorem 2 is an immediate application of a well known Theorem of Yorke (1970). In what follows J(t) denotes the Jacobian matrix $\frac{\partial \; fi}{\partial \; xj}$ and $J^T(t)$ its transpose.

2. STATEMENT OF RESULTS

Theorem 1. Suppose in (1.1) for some admissible control $u^* \in \Omega$,

(i) there exists a locally integrable function $\mu(t)$ such that the eigenvalues $\lambda_k(x,u^*,t)$ $(k= 1,2,...,n)$ of the symmetric matrix $\frac{1}{2}(J(t) + J^T(t))$ are less than or equal to $\mu(t)$, and satisfying

$$\lim_{t\to\infty} \int_{t_0}^{t} (\mu(s) + \lambda(s)) \, ds = -\infty, \lambda(t) = \max \lambda_k, \; (k=1,2...,n)$$

for all $(x,t) \in R^n \times I_0$;

(ii) The real in vector g satisfies

$$||g\,(x,u^*,t)|| \leq \mu(t)\; ||x|| + \gamma\,(t)\; ||x||^{\beta} \tag{2.1}$$

for all $(x,t) \in R^n \times I_0$, where $\gamma \geq 0$ is continuous functions of t and β is a constant in the range $0 \leq \beta < 1$.

(iii) The function μ in (2.1) satisfies

$$\int_{t1}^{t2} \mu(t)dt \leq \Delta_1 \cdot (t_2 - t_1) + \Delta_2 \tag{2.2}$$

for any t_2, t_1 with $t_1 \geq t_2 \geq t_0$, where $\Delta_i \geq 0$ $(i=1, 2)$ are constants.

Then there exists a constant $\varepsilon_1 = \varepsilon_1(\delta, A) > 0$ such that if $\Delta_1 \leq \varepsilon_1$ then corresponding to any constant $r > 0$ and any p in the range $1 \leq p \leq 2$, there is a constant $D_0 = D_0\,(r, \delta, A) > 0$ with the property that every solution $x(t)$ of (1.1) with $u(t) = u^*(t)$ satisfies

$$\sup_{t \geq t_0} ||x(t)|| \leq D_0 \sup_{t \geq t_0} \Big[\int_t^{t+r} \{||f(0,u^*(\tau),\tau)||^p + [\gamma\,(\tau)]^{p(1-\beta)^{-1}}\}\, dt\Big]^{1/p} \qquad (2.3)$$

provided that the latter supremum is finite.

The following Corollary follows at once from Theorem 1.

Corollary 1. Suppose in (1.1) that

(i) hypotheses (i) and (ii) of Theorem 1 hold;

(ii) one of the conditions

$$\int_{t_0}^{\infty} \mu(t)dt < \infty, \ \max_{t \geq T} \mu(t) \leq \varepsilon_1 < \alpha \text{ for some } t \geq t_0 \text{ holds}$$

(iii) $\int_t^{t+r} \{||f(0,u^*(\tau),\tau)||^p + [\gamma\,(\tau)]^{p(1-\beta)^{-1}}\}\, dt \to 0$ as $t \to \infty$

for some $r > 0$ and p $(1 \leq p \leq 2)$.

Then every solution $x(t)$ of (1.2) satisfies $x(t) \to 0$ as $t \to \infty$.

It is clear that when all the conditions of Corollary 1 hold then every solution $x(t)$ of (1.1) will, in finite time, approach a neighborhood of the origin. If (1.1) is now known to be locally controllable (near the origin) then C is the whole of R^n. The required condition for this is stated in Theorem 2 below.

Consider the related linear control system

$$\dot{x}(t) = J(t)x(t) + q(t), \ q(t) \varepsilon Q(t) \subseteq R^n \qquad (2.4)$$

where

$$J(t) = f_x(0,0,t) + g_x\,(0,0,t)$$

is an n x n matrix which is continuous in t. Here $Q(t) = K(\Lambda(t))$, where $K(\Lambda)$ is the unbounded closed convex cone of Λ with

$$\Lambda(t) = \{f(0,u,t) + g(0,u,t) : u \varepsilon \Omega\}$$
$$= f(0,\Omega,t) + g(0,\Omega,t);$$

that is, $K(\Lambda)$ is the smallest closed convex set containing Λ such that for all $c > 0$ $v \varepsilon K(\Lambda)$ implies $cv \varepsilon K(\Lambda)$.

Definition. Consider the linear system (2.4). The reachable set of (2.4) at time t is the set

$$IR(t,t_0) = \{\int_{t_0}^{t} X^{-1}(s,t_0)q(s)ds: \ q(t) \varepsilon Q(t) \text{ summable}\}$$

where $X(t,t_0)$ is the transition matrix (2.4).

Definition. The system (2.4) is proper on $[t_0,t]$, $t > t_0$, if and only if, the origin is an interior point of $IR(t,t_0)$. The system (2.4) is proper at t_0, if and only if, the origin is an interior point of $IR(t,t_0)$

for each $t > t_0$. If (2.4) is proper for each $t_0 \geq 0$ we simply say that the system is proper.

Theorem 2. Assume $0 \varepsilon \Omega$ and

$$f(0,0\ t) = g(0,0,t) = 0.$$

Suppose the linear system (2.4) is proper. Then (1.1) is locally controllable to the origin.

Theorem 3. Suppose in (1.1) for some admissible control $u^* \varepsilon \Omega$,

(i) there exists a locally integrable function $\mu(t)$ such that the eigenvalues $\lambda_k(x,u^*,t)$ $(k = 1,2,\dots,n)$ of the symmetric matrix $\frac{1}{2}(J(t) + J^T(t))$ are less than or equal to $\mu(t)$, and satisfying

$$\lim_{t\to\infty} \int_{t_0}^{t} (\mu(s) + \lambda(s))\, ds = -\infty, \quad \lambda(t) = \max \lambda_k, \ (k=1,2,..n)$$

for all $(x,t) \varepsilon R^n \times I_0$;

(ii) The real n vector g satisfies

$$||g(x,u^*,t)|| \leq \mu(t)\ ||x|| + \gamma(t)\ ||x||^{\beta} \tag{2.1}$$

for all $(x,t) \varepsilon R^n \times I_0$, where $\gamma \geq 0$ is continuous functions of t and β is a constant in the range $0 \leq \beta < 1$.

(iii) One of the conditions

$$\int_{t_0}^{\infty} \mu(t)dt < \infty, \quad \max \mu(t) \leq \varepsilon_1 \leq \infty \text{ for some } t \geq t_0$$

(iv) $$\int_t^{t+r} ||f(0,u^*,(\tau),\tau)||^p + [\gamma(\tau)]^{p(1-\beta)^{-1}} d\tau \to 0 \text{ as } t \to \infty$$

for some $\gamma > 0$ and p $(1 \leq p \leq 2)$;

(v) $0 \varepsilon \Omega$ and

$$f(0,0,t) = g(0,0,t) = 0;$$

and the system (2.4) is proper.

Then the domain C of null controllability of (1.2) is R^n.

3. PROOF OF THEOREM 1.

The proof adapts and extends the arguments in Ezeilo (1966), Chukwu (1974), and Sinha (1979) respectively

Let $x(t)$ be any solution of (1.1), and set

$$V(t) = (Px(t), x(t)),$$

where ( ) denotes the scalar product in R^n, and there exists a matrix P by condition (i) of Theorem 1. Because P is symmetric and positive definite

$$c||x(t)||^2 \leq V(t) \leq b||x(t||^2 \tag{3.1}$$

Here $b > 0$ and $c > 0$ are the greatest and least eigenvalues of P. Also from the symmetric nature of P,

$$\frac{dV}{dt} = 2(P\{f(x,u^*(t),t) + g(x,u^*(t),t)\},x)$$

$$= 2(Pf(x,u^*(t),t),x) + 2(Pg(x,u^*(t),t),x)$$

$$\equiv \quad W_1 \quad + \quad W_2,$$

say, for some $u(t) = u^*(t) \varepsilon \Omega$.

But it was proved in § 3 of Chukwu (1974) that under hypothesis (i)

$$W_1 \le -2\delta_0 V + D_4 V^{(1-\frac{1}{2}p)} ||f\ (0,u^*(t),t)||^P,\ D_4>0$$

for any constant p in the range $1 \le p \le 2$ where $\delta_0 \ge \frac{1}{2}(\ \mu(t) + \lambda(t)\ )b^{-1}$.

Furthermore because (2.1) holds there exists a constant $D_5 = D_5(A) > 0$ such that

$$|W_2| \le D_5\ (\mu(t)\ ||x||^2 + \gamma\ (t)\ ||x||^{1+\beta})\ ;$$

which on using (3.1) yields that

$$|W_2| \le D_6(\mu(t)V + \gamma\ (t)\ V^{\alpha_1}),$$

where $\alpha_1 = \frac{1}{2}(\beta+1)$ and $D_6 = D_5 \max\ (c^{-1},c^{-\alpha_1})$. It follows that

$$\frac{dV}{dt} \le -\ (\delta_0 - D_6\mu(t))V + D_4 V^{(1-\frac{1}{2}p)}\ ||f(0,u^*(t),t)||^P + W_3 \qquad (3.2)$$

where

$$W_3 = -\delta_0\ V + D_6\ \gamma\ (t)\ V^{\alpha_1}.$$

We now claim that for some $D_7 = D_7\ (\delta,A) > 0$

$$W_3 \le D_7 V^{(1-\frac{1}{2}p)} \quad \{\gamma\ (t)\}^{p/(1-\beta)} \qquad (3.3)$$

where p is as before: $1 \le p \le 2$. To prove this set $\upsilon = 1 - \frac{1}{2}p$ and note that υ is in the range $0 \le \upsilon \le \frac{1}{2} \le \alpha_1 < 1$. Clearly

$$W_3 = V^{\upsilon}\ V^{(\alpha_1 - \upsilon)} \quad \{D_6\ \gamma - \delta_0\ V^{(1-\alpha_1)}\}$$

$$\equiv V^{\upsilon}\ W_4\ .$$

Suppose $D_6\ \gamma \ge \delta_0\ V^{(1-\alpha_1)}$,

then

$$W_4 \le D_6 V^{(\alpha_1-\upsilon)} \quad \gamma_2$$

$$\le (D_6\ \delta_0^{-1})^{(\alpha_1-\upsilon)/(1-\alpha_1)}\ D_6\ \gamma^{1 + \{(\alpha_1-\upsilon)/(1-\alpha_1)\}}$$

$$= D\ \gamma^{(1-\upsilon)/(1-\alpha_1)}\ ,$$

where $D = D_6\ (D_6\ \delta_0^{-1})^{(\alpha_1-\upsilon)/(1-\alpha_1)}$

On the contrary, suppose

$$D_y\ \gamma \ < \ \delta_0\ V^{(1-\alpha_1)}\ .$$

Then obviously $W_4 \le 0$. It follows that

$$W_4 \leq D\,\gamma^{(1-\upsilon)/(1-\alpha_1)}$$

uniformly in t; so that

$$W_3 \leq D\,V^{\nu}\,\gamma^{(1-\mu)/(1-\alpha_1)} .$$

If we substitute $\upsilon = 1 - \frac{1}{2}p$, $\alpha_1 = \frac{1}{2}(1+\beta)$in the above inequality we obtain (3.3). Using (3.3) we have

$$\frac{dV}{dt} \leq -(\delta_0 - D_6\,\mu(t))V + 2\,D_8\,V^{(1-\frac{1}{2}p)}\{||f(0,u^*(t),t)||^p + [\gamma(t)]^{\frac{p}{1-\beta}}\},\ t \geq t_0$$

where $D_8 = \frac{1}{2}\max\,[D_4, D_7]$.

On substituting $V = W^{2/p}$ in the above inequality and on integrating the resulting differential inequality we obtain that for $t \geq t_0$

$$V^{\frac{1}{2}p} \leq V^{\frac{1}{2}p}(t_0)\exp\{-\tfrac{1}{2}p\,\Phi(t,t_0)\} + {} + p\,D_8\int_{t_0}^{t}\{||f(0,u^*(\tau),\tau)||^p + [\gamma(\tau)]^{p/(1-\beta)}\}\exp\{-\tfrac{1}{2}p\,\Phi(t,\tau)d\tau \tag{3.4}$$

where

$$\Phi(t_2,t_1) = \int_{t_1}^{t_2}\{\delta_0 - D_6\ \mu(t)\}\ dt.$$

Suppose μ in (1.3) satisfies (2.2) then clearly

$$\Phi(t_2, t_1) \geq (\delta_0 - D_6\,\Delta_1)(t_2 - t_1) - D_6\,\Delta_2,\ (t_2 \geq t_1);$$

so that if we further assume that

$$\Delta_1 \leq \tfrac{1}{2}\,\delta_0\,D_6^{-1} \tag{3.5}$$

then

$$\Phi(t_2, t_1) \geq \tfrac{1}{2}\,\delta_0\,(t_2 - t_1) - D_6\,\Delta_2. \tag{3.6}$$

It is now clear from (3.4) and (3.6) that

$$V^{\frac{1}{2}p} \leq D_9\,V^{\frac{1}{2}p}(t_0)\exp(-\tfrac{1}{2}p\,\delta_0\,(t-t_0)) + {} + p\,D_8\,D_9\int_{t_0}^{t}||f(0,u^*(\tau),\tau)||^p + [\gamma(\tau)]^{p/(1-\beta)}\exp -\{\tfrac{1}{2}p\delta_0(t-\tau)\}d\tau$$

for all $t \geq t_0$ where $D_9 = \exp(\frac{1}{2}p\,D_6\,\Delta_2)$.

Utilizing (3.1) once more we see that the above inequality implies that

$$||x(t)||^p \leq D_{10}||x(t_0)||^p\exp\{-\tfrac{1}{2}p\,\delta_0(t-t_0)\} + {} + D_{11}\int_{t_0}^{t}\{||f(0,u^*(\tau),\tau)||^p + [\gamma(\tau)]^{p\,(1-\beta)}\}\exp\{-\tfrac{1}{2}\,\delta_0 p(t-\tau)d\tau \tag{3.7}$$

where $D_{10} \equiv D_{10}(\delta, \Delta_2, A) > 0$, $D_{11} \equiv D_{11}(\delta, \Delta_2, A) > 0$

We shall next refine the inequality (3.7) to deduce (2.3). First set

$\lambda = \frac{1}{2} p \delta_0$.

Then from (3.7)

$$||x(t)||^p \le D_{12} e^{-\lambda t} + + D_{11}\int_{t_0}^{t} \{||f(0,u^*(\tau),\tau)||^p + [\gamma(\tau)]^{\frac{p}{1-\beta}} \exp\{-\lambda(t-\tau)\}d\tau \qquad (3.8)$$

for all $t \geq t_0$;

where $D_{12} = D(x(t_0), t_0, D_{10})$. Next suppose that

$$\Phi_0 = \sup_{t \geq t_0} \int_{t_0}^{t} \{||f(0,u^*(\tau),\tau)||^p + [\gamma(\tau)]^{p/(1-\beta)} < \infty \qquad (3.9)$$

Since (3.8) is the required analogue of (3.17) in Chukwu (1974) for our system (1.1), from this point on our arguments follow exactly as in § 3 of the cited reference.

4. PROOFS OF THEOREMS 2 AND 3

Proof of Theorem 2. The Theorem of Yorke (1970) is immediately applicable: The perturbed system (1.1) is locally controllable if and only if, the reachable set of (2.4) contains a neighborhood of the origin. Since this later condition on (2.4) is equivalent to the system being proper the proof of Theorem 2 is concluded: Hermes and LaSalle p. 78 (1969).

Proof of Theorem 3. Assume all the conditions of Corollary 1. Then there is a finite time T such that every solution $x(t)$ with $x(t_0) = x_0$ has $x(T)$ in an arbitrary neighborhood of the origin. Because of Theorem 2, (1.1) is locally controllable, and hence the point $x(T)$ is indeed steered to the origin in finite time, by some admissible control.

5. EXAMPLES

1. In (1.1) if $g(x,u,t) \equiv 0$ then hypotheses (ii) and (iii) of Theorem 1 and hypothesis (ii) of Corollary 1 are trivially verified. Corollary 1 then reduces to Theorem 1 of Chukwu (1974).
2. In (1.1) if $f(x,u,t) \equiv A_1 x + B_1 u$

   $g(x,u,t) \equiv A_2 x$

   where u is a continuous function with values in Ω, a cube in R^m, and where A_i is a constant n x n matrix and B_1 a constant n x m matrix, (i=1,2) then we have the following:

   The domain of null controllability of this special system is R^n if

(i) the eigenvalues of A_1 are all negative;

(ii) $\text{rank}\,[C, LC, \ldots L^{n-1} C] = n$

where $L = A_1 + A_2$ and $C = B_1$

Indeed, because the eigenvalues of the matrix A_1 are all negative hypothesis (i) of Theorem 1 is true with A = I, the identity matrix. Also

$$||A_2 x|| \le ||A_2|| \quad ||x||,$$

where the norm of the matrix $||A_2||$ is defined by

$$||A_2|| = \max_{\substack{1 \le i \le n \\ 1 \le j \le n}} |a_{ij}|$$

with a_{ij} the components of A_2. Thus we can take $\mu = ||A_2||$ and $\gamma = 0$. so that $\beta = 0$. Since A is constant hypothesis (ii) of Corollary 1 is trivially verified. It is clear from the nature of Ω that zero is an admissible control. Hence we can take $u^*(t) = 0$. Thus

$$f(0,u^*,t) = Bu^* = 0,$$

and

$$\int_t^{t+r} \{||f(0,u^*(\tau),(\tau)||^P + [\gamma(\tau)]^{P(1-\beta)^{-1}}\} d\tau = 0.$$

It follows that hypothesis (iii) of Corollary 1 is automatically true. Corollary 1 is now established for special system.

Next note that in the related linear equation,

$$L(t) = f_x(0,0,t) + g_x(0,0,t) = A_1 + A_2 \equiv L$$

$$\Lambda(t) = (B_1 \Omega) \equiv \Omega$$

But then a necessary and sufficient condition (Hermes and LaSalle (1969) p. 74) for the equation

$$\dot{x} = Lx + q(t) \qquad q(t) \subseteq K(C\Omega)$$

to be proper is that

$$\text{rank } [C, LC, \ldots, L^{n-1} C] = n.$$

Thus the conclusion of Theorem 2 holds for our system. It follows now from Theorem 3 that in the special system of example 2, conditions (i) and (ii) are sufficient for the domain of null controllability to be R^n.

REFERENCES

Chukwu, E. N. (1974), Finite Time Controllability of Nonlinear Control Processes, SIAM J. Control Vol., No., 1975, p. 807-816.

Ezeilo, J. O. C. (1966), An estimate for the solutions of a certain system of differential equations, Nigerian Journal of Science, Vol. 1, No. 1 pp. 5-10.

Hermes, H. and LaSalle, J. P., (1969), Functional Analysis and Time Optimal Control, Academic Press, New York.

Lukes, D. L. (1972), Global Controllability of Nonlinear Systems, SIAM J. Control 10, pp. 112-126.

Lukes, D. L. (1973), Erratum: Global Controllability of Nonlinear Systems, SIAM J. Control, 11 p. 186.

York, J. A. (1970), The Maximum Principle and Controllability of Nonlinear Equations, Technical Note BN-645, University of Maryland, 1F DAM, College Park, MD.

Sinha, A. S. C., (1979), Theory of Cone Methods for Null Controllability of Nonlinear Control Systems, IEEE Trans. on Automatic Control. Vol., AC-24 No. 4, 1979, pp. 628-630.

Sinha, A. S. C. (1978), Null Controllability of AMCD-Spacecraft Attitude Control System, Int. J. Systems Sci., Vol. 9, No. 2, pp. 205-217.

Anderson, W. W., Groom, N. J. and Wolley, C. T. (1979), Annular Suspension and Pointing System, J. Guidance and Control Vol. 2, No. 5, pp. 367-373.

Anderson, W. W. and Groom, N. J. (1975), The Annular Momentum Control Device (AMCD) and Potential Applications, NASA TN D-7866.

PERIODIC EQUILIBRIA FOR THE MATRIX RICCATI EQUATION WITH PERIODIC COEFFICIENTS*

MARK A. SHAYMAN
Department of Systems Science and Mathematics
Washington University
St. Louis, Missouri 63130

ABSTRACT

A complete description is provided for all the T-periodic solutions (periodic equilibria) of the matrix Riccati differential equation in which the coefficient matrices are T-periodic for some $T > 0$. Several key results for the equilibria of the time-invariant Riccati equation are generalized to the periodic equilibria of the periodic Riccati equation.

1. INTRODUCTION

By the Riccati differential equation (RDE) we mean the time-varying quadratic differential equation

$$\dot{K}(t) = -A'(t)K(t) - K(t)A(t) + K(t)B(t)B'(t)K(t) - C'(t)C(t)$$

defined on the vector space $S(n)$ of real symmetric $n \times n$ matrices. $A(t)$, $B(t)$, and $C(t)$ are real matrices of dimensions $n \times n$, $n \times m$, and $p \times n$ respectively. By the periodic Riccati differential equation (PRDE) we mean the special case of the RDE in which $A(t)$, $B(t)$, and $C(t)$ are each periodic with period $T > 0$ and are integrable over $[0,T]$. Due to its importance for the solution of the least squares control problem for a periodic linear system, the PRDE has been the subject of several investigations in the past few years, including [6] [1] [9] [3] [11] [12]. However, our understanding of the properties of the PRDE has remained considerably less advanced than our understanding of the properties of the time-invariant RDE.

The purpose of this paper is to generalize to the T-periodic solutions (periodic equilibria) of the PRDE several key results for the constant solutions (equilibria) of the time-invariant RDE. In particular, we establish the existence of maximal and minimal periodic equilibria, and prove a theorem which gives a complete description of the periodic equilibria in terms of the invariant subspaces of a certain monodromy matrix. This theorem generalizes to the PRDE a well-known result of J. C. Willems [15] which describes all of the equilibria for the time-invariant RDE.

The following notation will be used: If X is any matrix, Sp X denotes the subspace spanned by the columns of X. If X is a square matrix, Spec(X) denotes the spectrum of X. Also $L^+(X)$ $(L^-(X))$ denotes the invariant subspace associated with the eigenvalues of X inside (outside) the unit circle. If S is any subspace of $\mathbb{R}^{2n}$, $S^\perp$ denotes the orthogonal complement of S using the standard inner product on $\mathbb{R}^{2n}$. We use $\psi_A(t,t_0)$ to denote the transition matrix for the linear differential equation $\dot{x}(t) = A(t)x(t)$. Some proofs are omitted due to space limitations. They are available in [14].

2. BACKGROUND

Associated with the RDE is the $2n \times 2n$ Hamiltonian matrix

---

*Research partially supported by National Science Foundation under Grant ECS-8301015.

$$H(t) = \begin{bmatrix} A(t) & -B(t)B'(t) \\ -C'(t)C(t) & -A'(t) \end{bmatrix}.$$

Let $\Phi(t,t_0)$ denote the transition matrix corresponding to $H(t)$. Partition $\Phi(t,t_0)$ into n x n blocks as

$$\phi(t,t_0) = \begin{bmatrix} \phi_{11}(t,t_0) & \phi_{12}(t,t_0) \\ \phi_{21}(t,t_0) & \phi_{22}(t,t_0) \end{bmatrix}.$$

Let $K(t,K_0,t_0)$ denote the solution of the RDE which goes through K_0 at time t_0. It is well-known [4, p. 156] that

$$K(t,K_0,t_0) = [\phi_{21}(t,t_0) + \phi_{22}(t,t_0)K_0][\phi_{11}(t,t_0) + \phi_{12}(t,t_0)K_0]^{-1}. \quad (1)$$

This formula is easily verified by differentiation. It is valid as long as the inverse exists. The matrix $\phi_{11}(t,t_0) + \phi_{12}(t,t_0)K_0$ becoming singular at $t = t_1$ is equivalent to the solution $K(t,K_0,t_0)$ having a finite escape time at $t = t_1$.

In order to study the RDE, it is useful to compactify the phase space. The natural compactification of the phase space S(n) for the RDE has been described by R. Hermann and C. Martin [7] [10] and is the so-called Lagrange-Grassmann manifold $L(n)$. It is defined as follows: Let J denote the 2n x 2n matrix $[\begin{smallmatrix} 0 & I \\ -I & 0 \end{smallmatrix}]$ and define a skew-symmetric bilinear form ω on $\mathbb{R}^{2n}$ by $\omega(x,y) = x'Jy$. Let $G^n(\mathbb{R}^{2n})$ denote the Grassmann manifold of all n-dimensional subspaces of $\mathbb{R}^{2n}$. Then $L(n) = \{S \in G^n(\mathbb{R}^{2n}): \omega(x,y) = 0, \forall x, y \in S\}$. Thus, $L(n)$ consists of those n-dimensional subspaces of $\mathbb{R}^{2n}$ on which ω vanishes identically.

It is natural to view $L(n)$ as a compactification of S(n). To see this, we note that the n-dimensional subspace $Sp[\begin{smallmatrix} I \\ K \end{smallmatrix}]$ (the column space of the 2n x n matrix $[\begin{smallmatrix} I \\ K \end{smallmatrix}]$) belongs to $L(n)$ if and only if the n x n matrix K is symmetric. Thus, we can define a mapping $\gamma : S(n) \to L(n)$ by $\gamma(K) = Sp[\begin{smallmatrix} I \\ K \end{smallmatrix}]$. Let $L_0(n)$ consist of those subspaces in $L(n)$ which are complementary to the n-dimensional subspace $Sp[\begin{smallmatrix} 0 \\ I \end{smallmatrix}]$. In other words, $L_0(n) = \{S \in L(n) : S \cap Sp[\begin{smallmatrix} 0 \\ I \end{smallmatrix}] = 0\}$. It is easy to show that γ is an embedding of S(n) into $L(n)$ with image $L_0(n)$. Since $L(n)$ is compact and $L_0(n)$ is an open and dense subset of $L(n)$, we can use the embedding γ to identify S(n) with $L_0(n)$ and thereby regard $L(n)$ as a compactification of S(n).

Define a time-varying flow on $L(n)$ by $S(t,S_0,t_0) = \Phi(t,t_0)(S_0)$, the image of the subspace S_0 under the nonsingular linear mapping $\Phi(t,t_0)$. Note that since H(t) is a Hamiltonian matrix $(JH(t) + H'(t)J = 0)$, it follows that $\Phi(t,t_0)$ is symplectic $(\Phi'(t,t_0)J\Phi(t,t_0) = J)$. This implies that if $S_0 \in L(n)$, then $\Phi(t,t_0)(S_0) \in L(n)$, so $S(t,S_0,t_0)$ is indeed a flow on $L(n)$.

If $K_0 \in S(n)$, then

$$S(t,\gamma(K_0),t_0) = \mathrm{Sp}\Phi(t,t_0) \begin{bmatrix} I \\ K_0 \end{bmatrix} = \mathrm{Sp}\begin{bmatrix} \Phi_{11}(t,t_0)+\Phi_{12}(t,t_0)K_0 \\ \Phi_{21}(t,t_0)+\Phi_{22}(t,t_0)K_0 \end{bmatrix} .$$

From (1), we have

$$\gamma(K(t,K_0,t_0)) = \mathrm{Sp}\begin{bmatrix} I \\ [\phi_{21}(t,t_0)+\phi_{22}(t,t_0)K_0][\phi_{11}(t,t_0)+\phi_{12}(t,t_0)K_0]^{-1} \end{bmatrix}$$
$$= \mathrm{Sp}\begin{bmatrix} \phi_{11}(t,t_0)+\phi_{12}(t,t_0)K_0 \\ \phi_{21}(t,t_0)+\phi_{22}(t,t_0)K_0 \end{bmatrix}$$

which is valid as long as the solution $K(t,K_0,t_0)$ continues to exist. Thus, the flow of the RDE on $S(n)$ is related to the flow which was defined on $\mathcal{L}(n)$ by the equation

$$\gamma(K(t,K_0,t_0)) = S(t,\gamma(K_0),t_0) \tag{2}$$

as long as $K(t,K_0,t_0)$ exists, or equivalently, as long as $S(t,\gamma(K_0),t_0)$ remains in the subset $\mathcal{L}_0(n)$ of $\mathcal{L}(n)$. In other words, (2) says that if we use the embedding γ to identify $S(n)$ with $\mathcal{L}_0(n)$, then the restriction to $\mathcal{L}_0(n)$ of the flow $S(t,S_0,t_0)$ is identified with the flow of the RDE. Thus, the embedding γ permits us to view $\mathcal{L}(n)$ as $S(n)$ together with some points added at infinity to obtain a compact space, and to view $S(t,S_0,t_0)$ as the extension of the flow of the RDE to the enlarged space.

3. PERIODIC EQUILIBRIA FOR THE PERIODIC RICCATI DIFFERENTIAL EQUATION

Throughout this section, it is assumed that the matrices $A(\cdot)$, $B(\cdot)$, and $C(\cdot)$ in the RDE are each periodic with period $T > 0$, and are integrable on $[0,T]$. The first result relates periodic solutions to invariant subspaces of $\Phi(t_0+T,t_0)$.

Lemma 1: (a) If $K(t,K_0,t_0)$ is a periodic equilibrium of the PRDE, then $\gamma(K_0)$ is $\Phi(t_0+T,t_0)$-invariant and $S(t,\gamma(K_0),t_0) \in \mathcal{L}_0(n)$, $\forall t$. (b) If S_0 is $\Phi(t_0+T,t_0)$-invariant and $S(t,S_0,t_0) \in \mathcal{L}_0(n)$, $\forall t$, then $K(t,\gamma^{-1}(S_0),t_0)$ $(=\gamma^{-1}(S(t,S_0,t_0)))$ is a periodic equilibrium of the PRDE.

Pf: (a) Suppose that $K(t,K_0,t_0)$ is a periodic equilibrium of the PRDE. Then $K(t,K_0,t_0)$ exists for all t, which implies (see (2)) that $S(t,\gamma(K_0),t_0) \in \mathcal{L}_0(n)$, $\forall t$. Since $K(t_0+T,K_0,t_0) = K_0$, (2) implies that $\gamma(K_0) = \gamma(K(t_0+T,K_0,t_0)) = S(t_0+T,\gamma(K_0),t_0) = \Phi(t_0+T,t_0)(\gamma(K_0))$, so $\gamma(K_0)$ is $\Phi(t_0+T,t_0)$-invariant. (b) Suppose that S_0 is $\Phi(t_0+T,t_0)$-invariant and $S(t,S_0,t_0) \in \mathcal{L}_0(n)$, $\forall t$. Then (2) gives the equation

$$\gamma(K(t,\gamma^{-1}(S_0),t_0)) = S(t,S_0,t_0)$$

which is valid for all t since $S(t,S_0,t_0)$ remains in $\mathcal{L}_0(n)$. ($\gamma^{-1}(S_0)$ is defined since $S_0 \in \mathcal{L}_0(n)$.) Hence, $K(t,\gamma^{-1}(S_0),t_0)$ exists for all t and

$$K(t,\gamma^{-1}(S_0),t_0) = \gamma^{-1}(S(t,S_0,t_0)).$$

Thus, $K(t+T,\gamma^{-1}(S_0),t_0) = \gamma^{-1}(S(t+T,S_0,t_0)) = \gamma^{-1}(\Phi(t+T,t_0)(S_0)) = \gamma^{-1}(\Phi(t+T,t_0+T)\Phi(t_0+T,t_0)(S_0)) = \gamma^{-1}(\Phi(t,t_0)(S_0)) = \gamma^{-1}(S(t,S_0,t_0)) = K(t,\gamma^{-1}(S_0),t_0)$. Thus, $K(t,\gamma^{-1}(S_0),t_0)$ is a periodic equilibrium. □

If $K(t)$ is a periodic solution of the PRDE, then $\gamma(K(t_0))$ is a $\Phi(t_0+T,t_0)$-invariant subspace. The next result describes the relationship between the restriction $\Phi(t_0+T,t_0)|\gamma(K(t_0))$ and the transition matrix $\psi_{A-BB'K}(t,\tau)$ for the periodic closed-loop system $A(t) - B(t)B'(t)K(t)$. The simple proof is omitted.

Lemma 2: Suppose that $K(t)$ is a periodic equilibrium of the PRDE. Then $\psi_{A-BB'K}(t_0+T,t_0)$ is the matrix for $\Phi(t_0+T,t_0)|\gamma(K(t_0))$ with respect to the basis consisting of the columns of $\begin{bmatrix} I \\ K(t_0)\end{bmatrix}$.

Corollary: Suppose that $K(t)$ is a periodic equilibrium of the PRDE. Then the characteristic multipliers of the periodic closed-loop system $A(t) - B(t)B'(t)K(t)$ are the same as the eigenvalues of $\Phi(t_0+T,t_0)|\gamma(K(t_0))$.

Lemma 1 shows that if S_0 is a $\Phi(t_0+T,t_0)$-invariant subspace which belongs to $L(n)$, then $S(t,S_0,t_0)$ corresponds to a periodic equilibrium $K(t) = \gamma^{-1}(S(t,S_0,t_0))$ provided that $S(t,S_0,t_0) \in L_0(n)$ for all t. Thus, the existence of periodic equilibria for the PRDE is related to the existence of $\Phi(t_0+T,t_0)$-invariant subspaces which belong to $L(n)$. The next lemma will be useful in establishing the existence of such subspaces. Its proof involves standard linear algebra and is omitted.

Lemma 3: Let M be a symplectic matrix. Suppose that S_1 and S_2 are each sums of primary components of M such that if $\lambda \in \mathrm{Spec}(M|S_1)$, then $\lambda^{-1} \notin \mathrm{Spec}(M|S_2)$. Then $x'Jy = 0$, $\forall x \in S_1$, $\forall y \in S_2$.

As a preliminary to obtaining further results regarding the periodic equilibria of the PRDE, we consider a special case. By the stable homogeneous periodic Riccati differential equation (SHPRDE), we mean the special case of the PRDE where $C(t) \equiv 0$ and all of the characteristic multipliers of $A(\cdot)$ are inside the unit circle. We use $H(t)$ to denote the Hamiltonian matrix $\begin{bmatrix} A(t) & -B(t)B'(t) \\ 0 & -A'(t)\end{bmatrix}$ for the SHPRDE, and we use $\hat{\Phi}(t,t_0)$ to denote the transition matrix corresponding to $H(t)$. Let $W_A(t_0,t)$ denote the controllability Gramian for the pair $(A(\cdot),B(\cdot))$. In other words,

$$W_A(t_0,t) = \int_{t_0}^{t} \psi_A(t_0,\sigma)B(\sigma)B'(\sigma)\psi_A'(t_0,\sigma)\,d\sigma.$$

It may be verified directly that

$$\hat{\Phi}(t,t_0) = \begin{bmatrix} \psi_A(t,t_0) & -\psi_A(t,t_0)W_A(t_0,t) \\ 0 & \psi_A'(t_0,t)\end{bmatrix} \tag{3}$$

We will need the following standard result concerning the discrete version of the (algebraic) Liapunov equation.

Lemma 4: Suppose that the symmetric matrix Z satisfies the equation

$$Z - R'ZR = -L$$

where $L \geq 0$ and $|\lambda| < 1$, $\forall\lambda \in \mathrm{Spec}(R)$. Then $Z \leq 0$.

Lemma 5: Suppose that $S \in L(n)$ and that S is $\hat{\Phi}(t_0+T,t_0)$-invariant. Let $\begin{bmatrix} X \\ Y \end{bmatrix}$ be any basis for S. Then $X'Y \leq 0$.

Pf: Note that since $S \in L(n)$, $X'Y$ is symmetric. We begin by considering the special case where $S = L^-(\hat{\Phi}(t_0+T,t_0))$. In other words, S is the subspace corresponding to the eigenvalues of $\hat{\Phi}(t_0+T,t_0)$ which are outside the unit circle. It is clear from (3) that $\hat{\Phi}(t_0+T,t_0)$ has no eigenvalues on the unit circle, so S is an n-dimensional $\hat{\Phi}(t_0+T,t_0)$-invariant subspace. Applying Lemma 3 with $S_1 = S_2 = S$, we conclude that $S \in L(n)$.

Let $\begin{bmatrix} \hat{X} \\ \hat{Y} \end{bmatrix}$ be a basis for S. Let $N = \psi_A(t_0+T,t_0)$ and let $W = W_A(t_0,t_0+T)$. From (3), we have

$$\hat{\Phi}(t_0+T,t_0) = \begin{bmatrix} N & -NW \\ 0 & (N^{-1})' \end{bmatrix}$$

Since S is $\hat{\Phi}(t_0+T,t_0)$-invariant, there exists an n x n matrix R with $|\lambda| > 1$, $\forall\lambda \in \mathrm{Spec}(R)$ such that

$$\begin{bmatrix} N & -NW \\ 0 & (N^{-1})' \end{bmatrix} \begin{bmatrix} \hat{X} \\ \hat{Y} \end{bmatrix} = \begin{bmatrix} \hat{X} \\ \hat{Y} \end{bmatrix} R.$$ Letting $P = \hat{Y}'\hat{X}$, it is easy to obtain

$$P - (R^{-1})'PR^{-1} = -(R^{-1})'\hat{Y}'W\hat{Y}R^{-1}.$$

Since $W \geq 0$, we conclude from Lemma 4 that $P \leq 0$. Thus, $\hat{X}'\hat{Y} = \hat{Y}'\hat{X} \leq 0$.

We continue to use $\begin{bmatrix} \hat{X} \\ \hat{Y} \end{bmatrix}$ to denote a basis for $L^-(\hat{\Phi}(t_0+T,t_0))$, but now we let S denote an arbitrary subspace in $L(n)$ which is $\hat{\Phi}(t_0+T,t_0)$-invariant. Then $S = [S \cap L^+(\hat{\Phi}(t_0+T,t_0))] \oplus [S \cap L^-(\hat{\Phi}(t_0+T,t_0))] = [S \cap \mathrm{Sp}\begin{bmatrix} I \\ 0 \end{bmatrix}] \oplus [S \cap \mathrm{Sp}\begin{bmatrix} \hat{X} \\ \hat{Y} \end{bmatrix}]$. Let $\ell = \dim S \cap \mathrm{Sp}\begin{bmatrix} I \\ 0 \end{bmatrix}$. There exist $n \times \ell$ and $n \times (n-\ell)$ full rank matrices C, D such that $S \cap \mathrm{Sp}\begin{bmatrix} I \\ 0 \end{bmatrix} = \mathrm{Sp}\begin{bmatrix} I \\ 0 \end{bmatrix}C$ and $S \cap \mathrm{Sp}\begin{bmatrix} \hat{X} \\ \hat{Y} \end{bmatrix} = \mathrm{Sp}\begin{bmatrix} \hat{X} \\ \hat{Y} \end{bmatrix}D$. So

$$S = \mathrm{Sp}\begin{bmatrix} C & \hat{X}D \\ 0 & \hat{Y}D \end{bmatrix}.$$

Let $X = [C \;\; \hat{X}D]$ and let $Y = [0 \;\; \hat{Y}D]$. Since $S \in L(n)$, $X'Y$ must be symmetric, which implies that $C'\hat{Y}D = 0$. Then we obtain

$$X'Y = \begin{bmatrix} 0 & 0 \\ 0 & D'\hat{X}'\hat{Y}D \end{bmatrix}.$$

Since $\hat{X}'\hat{Y} \leq 0$, we conclude that $X'Y \leq 0$. ☐

Corollary: Let K(t) be a periodic equilibrium of the SHPRDE. Then $K(t) \leq 0$.

Pf: Given any t, we have $K(t) = K(t+T)$. Thus, $\mathrm{Sp}\begin{bmatrix} I \\ K(t) \end{bmatrix}$ is $\hat{\Phi}(t+T,t)$-invariant. By Lemma 5, $K(t) \leq 0$. $\square$

Lemma 6: Suppose that $(A(\cdot),B(\cdot))$ is controllable. Then every subspace belonging to $\mathcal{L}(n)$ which is $\Phi(t_0+T,t_0)$-invariant belongs to $\mathcal{L}_0(n)$.

Pf: Let $S \in \mathcal{L}(n)$ and suppose that S is $\hat{\Phi}(t_0+T,t_0)$-invariant. Let $\begin{bmatrix} X \\ Y \end{bmatrix}$ be a basis matrix for S. Since $\hat{\Phi}(t_0-nT,t_0) = [\hat{\Phi}(t_0+T,t_0)]^{-n}$, S is $\hat{\Phi}(t_0-nT,t_0)$-invariant. Hence, there exists an n x n matrix R such that

$$\hat{\Phi}(t_0-nT,t_0)\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} X \\ Y \end{bmatrix} R.$$

From (3) we have

$$\psi_A(t_0-nT,t_0)X - \psi_A(t_0-nT,t_0)W_A(t_0,t_0-nT)Y = XR \tag{4}$$

$$\psi_A(t_0,t_0-nT)Y = YR. \tag{5}$$

From (4) and (5), it is not hard to show that

$$\psi_A(t_0-nT,t_0)X + W_A(t_0-nT,t_0)YR = XR. \tag{6}$$

Suppose that there exists $v \neq 0$ with $Xv = 0$. Premultiplying (6) by $v'R'Y'$ and postmultiplying by v gives

$$v'R'Y'W_A(t_0-nT,t_0)YRv = v'R'Y'XRv. \tag{7}$$

Since $(A(\cdot),B(\cdot))$ is controllable, it follows from [5, Proposition 3] that $W_A(t_0-nT,t_0)$ is positive definite. By Lemma 5, $Y'X$ is negative semidefinite. We conclude from (7) that $YRv = 0$. Then (5) implies that $Yv = 0$. But this is impossible since $Xv = 0$ and $\begin{bmatrix} X \\ Y \end{bmatrix}$ has rank n. Hence, X must be nonsingular. $\square$

We are now prepared to describe all of the periodic equilibria for the PRDE using the above results for the SHPRDE as tools. Recall that $L^+(\Phi(T,0))$ is the $\Phi(T,0)$-invariant subspace associated with the eigenvalues of $\Phi(T,0)$ inside the unit circle. If $\Phi(T,0)$ has no eigenvalues on the unit circle. $L^+(\Phi(T,0))$ is n-dimensional. In this case, it follows from Lemma 3 that $L^+(\Phi(T,0)) \in \mathcal{L}(n)$. If $\Phi(t,0)(L^+(\Phi(T,0))) \in \mathcal{L}_0(n)$, $\forall t$, then it follows from Lemma 1 that $K(t,\gamma^{-1}(L^+(\Phi(T,0))),0)$ is a periodic equilibrium. If this solution exists, we will denote it by $K^+(t)$. It follows from the corollary to Lemma 2 that if it exists, $K^+(t)$ is the unique periodic equilibrium with the property that every characteristic multiplier of the associated closed-loop system is inside the unit circle.

If $\Phi(T,0)$ has no eigenvalues on the unit circle and $\Phi(t,0)(L^-(\Phi(T,0))) \in \mathcal{L}_0(n)$, $\forall t$, then another periodic equilibrium is given by $K(t,\gamma^{-1}(L^-(\Phi(T,0))),0)$. We denote this solution by $K^-(t)$, If it exists, $K^-(t)$ is the unique periodic equilibrium with the property that every characteristic multiplier of the associated closed-loop system is outside the unit circle.

The next result shows that if it exists $K^+(t)$ is the maximal periodic equilibrium (in the partial ordering on symmetric matrices). This is true even if $(A(\cdot),B(\cdot))$ is not controllable.

Theorem 1: Suppose that $\Phi(T,0)$ has no eigenvalues on the unit circle and that $K^{+}(t)$ exists. If $K(t)$ is any other periodic equilibrium, then $K(t) \leq K^{+}(t)$.

Pf: Let $A^{+}(t) = A(t) - B(t)B'(t)K^{+}(t)$ and consider the SHPRDE

$$\dot{\hat{K}}(t) = -A^{+\prime}(t)\hat{K}(t) - \hat{K}(t)A^{+}(t) + \hat{K}(t)B(t)B'(t)\hat{K}(t).$$

The associated Hamiltonian matrix is

$$\hat{H}(t) = \begin{bmatrix} A^{+}(t) & -B(t)B'(t) \\ 0 & -A^{+\prime}(t) \end{bmatrix}.$$

Let $\hat{\Phi}(t,t_0)$ denote the transition matrix corresponding to $\hat{H}(t)$. Let $K(t)$ be a periodic equilibrium of the PRDE. It is straightforward to check that $K(t) - K^{+}(t)$ is a periodic equilibrium of the SHPRDE. By Lemma 1, the subspace $\mathrm{Sp}\begin{bmatrix} I \\ K(t)-K^{+}(t) \end{bmatrix}$ is $\hat{\Phi}(t+T,t)$-invariant. By Lemma 5, $K(t) - K^{+}(t) \leq 0$ which completes the proof. □

We will need to use the following known result which is an immediate consequence of [9, Theorem 4].

Theorem 2: Suppose that $(A(\cdot),B(\cdot))$ is controllable and that $\Phi(T,0)$ has no eigenvalues on the unit circle. Then the periodic equilibrium $K^{+}(t)$ of the PRDE exists.

The next result is crucial in order to establish the existence of periodic equilibria in addition to $K^{+}(t)$. It generalizes to the PRDE the result in Lemma 6 for the SHPRDE.

Lemma 7: Suppose that $(A(\cdot),B(\cdot))$ is controllable and that $\Phi(T,0)$ has no eigenvalues on the unit circle. Then every subspace belonging to $L(n)$ which is $\Phi(t_0+T,t_0)$-invariant belongs to $L_0(n)$.

Pf: Let $S \in L(n)$ and suppose that S is $\Phi(t_0+T,t_0)$-invariant. $K^{+}(t)$ exists by Theorem 2. Let $U(t) = \begin{bmatrix} I & 0 \\ -K^{+}(t) & I \end{bmatrix}$ and let $\hat{H}(t)$ and $\hat{\Phi}(t,t_0)$ be defined as in the proof of Theorem 1. It is straightforward to check that $\hat{H}(t) = \dot{U}(t)U^{-1}(t) + U(t)H(t)U^{-1}(t)$ and $\hat{\Phi}(t,t_0) = U(t)\Phi(t,t_0)U^{-1}(t_0)$. Since $U(t)$ is periodic, we have

$$\hat{\Phi}(t_0+T,t_0) = U(t_0)\Phi(t_0+T,t_0)U^{-1}(t_0). \tag{8}$$

Since $U(t_0)$ is symplectic, $U(t_0)(S) \in L(n)$. From (8) it follows that $U(t_0)(S)$ is $\hat{\Phi}(t_0+T,t_0)$-invariant. By Lemma 6, $U(t_0)(S) \in L_0(n)$. Let $\begin{bmatrix} X \\ Y \end{bmatrix}$ be a basis for S. Then $U(t_0)(S) = \mathrm{Sp}\begin{bmatrix} X \\ -K^{+}(t_0)X+Y \end{bmatrix}$. Since $U(t_0)(S) \in L_0(n)$, X is nonsingular. But this implies that $S \in L_0(n)$. □

Lemma 7 is a key result. It shows that if $(A(\cdot),B(\cdot))$ is controllable and $\Phi(T,0)$ has no eigenvalues on the unit circle, then every periodic orbit of the flow on $L(n)$ is completely contained in $L_0(n)$ and hence corresponds to a periodic orbit of the PRDE on $S(n)$. On the other hand, there is a one-to-one correspondence between periodic orbits of the flow on $L(n)$ and the set of those elements of $L(n)$ which are $\Phi(T,0)$-invariant. Hence, there is a one-to-one correspondence between the set of $\Phi(T,0)$-invariant elements of $L(n)$ and the set of periodic equilibria of the PRDE. The explicit form of this correspondence is given by the next result whose proof is an easy consequence of Lemmas 1 and 7.

Theorem 3: Suppose that $(A(\cdot),B(\cdot))$ is controllable and $\Phi(T,0)$ has no eigenvalues on the unit circle. Each $\Phi(T,0)$-invariant subspace $S_0 \varepsilon L(n)$ determines a unique periodic equilibrium of the PRDE which is given by $\gamma^{-1}(\Phi(t,0)(S_0))$. Furthermore, every periodic equilibrium is obtained in this way.

This result describes every periodic equilibrium of the PRDE. However, the description is given in terms of the set of $\Phi(T,0)$-invariant subspaces which belong to $L(n)$. Consequently, the next task is to describe this set. The proof of the following result is similar to the proof of [13, Theorem 2].

Lemma 8: Suppose that $\Phi(T,0)$ has no eigenvalues on the unit circle. Then the mapping $S \xmapsto{\delta} S \cap L^+(\Phi(T,0))$ is a bijection of the set of $\Phi(T,0)$-invariant subspaces which belong to $L(n)$ onto the set of $\Phi(T,0)$-invariant subspaces of $L^+(\Phi(T,0))$. If S_1 is a $\Phi(T,0)$-invariant subspace of $L^+(\Phi(T,0))$, then $\delta^{-1}(S_1) = S_1 \oplus ([J(S_1)]^\perp \cap L^-(\Phi(T,0)))$.

Using this result, we immediately obtain a refinement of Theorem 3.

Theorem 4: Suppose that $(A(\cdot),B(\cdot))$ is controllable and $\Phi(T,0)$ has no eigenvalues on the unit circle. Each $\Phi(T,0)$-invariant subspace of $L^+(\Phi(T,0))$ determines a unique periodic equilibrium of the PRDE. If S_1 is such a subspace, the corresponding periodic equilibrium is given by $\gamma^{-1}(\Phi(t,0)(S_1 \oplus ([J(S_1)]^\perp \cap L^-(\Phi(T,0)))))$. Furthermore, every periodic equilibrium of the PRDE is obtained in this way.

This result is an improvement over Theorem 3 because it describes the periodic equilibria in terms of the invariant subspaces of $\Phi(T,0)|L^+(\Phi(T,0))$, which is a set of known structure. On the other hand, Theorem 3 describes the periodic equilibria in terms of the set of n-dimensional $\Phi(T,0)$-invariant subspaces which satisfy an additional condition, namely that $[J(S)]^\perp = S$.

Theorem 4 gives a one-to-one correspondence between the set of periodic equilibria and the set of $\Phi(T,0)$-invariant subspaces of $L^+(\Phi(T,0))$. By Lemma 2, $\psi_{A^+}(T,0)$ is the matrix for $\Phi(T,0)|L^+(\Phi(T,0))$ with respect to the basis given by the columns of $\begin{bmatrix} I \\ K^+(0) \end{bmatrix}$. Thus, there is a one-to-one correspondence between the set of periodic equilibria and the set of $\psi_{A^+}(T,0)$-invariant subspaces of $\mathbb{R}^n$. Using this fact, we will prove a very useful theorem which shows how every periodic equilibrium can be constructed from the pair $K^+(t)$ and $K^-(t)$.

We need two preliminary results.

Lemma 9: Suppose that $(A(\cdot),B(\cdot))$ is controllable and $\Phi(T,0)$ has no eigenvalues on the unit circle. Let $\Delta(t) = K^+(t) - K^-(t)$. Then $\Delta(t) > 0$, $\forall t$.

Pf: Since $(A(\cdot),B(\cdot))$ is controllable and $\Phi(T,0)$ has no eigenvalues on the unit circle, it follows from Theorem 3 that both $K^+(t)$ and $K^-(t)$ exist. By Theorem 1, $\Delta(t) \geq 0$. It is trivial that $\Delta(t)$ is singular if and only if $\gamma(K^+(t)) \cap \gamma(K^-(t)) \neq 0$. However, it follows easily from the definitions of

$K^+(t)$ and $K^-(t)$ that $\gamma(K^+(t)) = L^+(\Phi(t+T,t))$ and $\gamma(K^-(t)) = L^-(\Phi(t+T,t))$, which implies that $\gamma(K^+(t)) \cap \gamma(K^-(t)) = 0$. Hence, $\Delta(t) > 0$. □

Lemma 10: Suppose that $(A(\cdot),B(\cdot))$ is controllable and that $\Phi(T,0)$ has no eigenvalues on the unit circle. Let $A^-(t) = A(t) - B(t)B'(t)K^-(t)$. Then

$$\psi'_{A^-}(t,t_0)\Delta(t)\psi_{A^+}(t,t_0) = \Delta(t_0), \ \forall t, \ t_0.$$

Pf: The equation holds trivially for $t = t_0$, and it is easy to check that the derivative of the lefthand side is identically zero. □

The following theorem shows how every periodic equilibrium can be constructed from $K^+(t)$ and $K^-(t)$. It generalizes to the periodic RDE a well known result of J. C. Willems [15] for the time-invariant RDE. The proof uses Theorem 4 and Lemmas 9 and 10 and is omitted due to its length.

Theorem 5: Suppose that $(A(\cdot),B(\cdot))$ is controllable and $\Phi(T,0)$ has no eigenvalues on the unit circle. There is a one-to-one correspondence between the set of periodic equilibria of the PRDE and the set of $\psi_{A^+}(T,0)$-invariant subspaces of $\mathbb{R}^n$. Let M_0 be a $\psi_{A^+}(T,0)$-invariant subspace, and let $M(t) = \psi_{A^+}(t,0)(M_0)$, and let $\Pi(t)$ be the projection onto $M(t)$ along $\Delta^{-1}(t)(M(t)^\perp)$. The periodic equilbrium corresponding to M_0 is

$$K(t) = K^+(t)\Pi(t) + K^-(t)[I-\Pi(t)].$$

Furthermore, $L^+(\psi_{A-BB'K}(t+T,t)) = M(t)$ and $L^-(\psi_{A-BB'K}(t+T,t)) = \Delta^{-1}(t)(M(t)^\perp)$.

As an easy consequence of Theorem 5, we have

Corollary: Suppose that $(A(\cdot),B(\cdot))$ is controllable and $\Phi(T,0)$ has no eigenvalues on the unit circle. Let $K(t)$ be any periodic equilibrium of the PRDE. Then $K^-(t) \leq K(t) \leq K^+(t)$, $\forall t$.

REFERENCES

1. E. Bekir and R. S. Bucy, "Periodic equilibria for matrix Riccati equations", Stochastics, 2 (1976), 1-104.
2. S. Bittanti et al., "H-controllability and observability of linear periodic systems", Relazione interna 82-16, Laboratorio di Controlli Automatici, Dipartimento di Elettronica del Politecnico di Milano.
3. S. Bittanti et al., "Periodic solutions of periodic Riccati equations", to appear in IEEE Trans. Automat. Contr.
4. R. W. Brockett, Finite Dimensional Linear Systems, John Wiley, New York, 1970.
5. P. Brunovsky, "Controllability and linear closed-loop controls in linear periodic systems", J. Differential Equations, 6 (1969), 296-313.
6. J. R. Canabal, "Periodic geometry of the Riccati equation", Stochastics, 1 (1974), 432-440.
7. R. Hermann, Cartanian Geometry, Nonlinear Waves, and Control Theory, Part A, Interdisciplinary Mathematics, Vol. XX, Math. Sci. Press, Brookline, MA, 1979.
8. G. A. Hewer, "Periodicity, detectability and the matrix Riccati equation", SIAM J. Contr., 13 (1975), 1235-1251.
9. H. Kano and T. Nishimura, "Periodic solutions of matrix Riccati equations with detectability and stabilizability", Int. J. Contr., 29 (1979), 471-487.
10. C. Martin, "Grassmannian manifolds, Riccati equations and feedback invariants of linear systems", in Geometrical Methods for the Theory of Linear Systems (ed. C. Byrnes and C. Martin), Reidel, Dordrecht, 1980.

11. D. A. Sanchez, "A note on periodic solutions of Riccati-type equations", SIAM J. Appl. Math., 17 (1969), 957-959.
12. D. A. Sanchez, "Computing periodic solutions of Riccati differential equations", Appl. Math. and Comp., 6 (1980), 283-287.
13. M. A. Shayman, "On the periodic solutions of the matrix Riccati equation", Math. Systems Theory, 16 (1983), 267-287.
14. M. A. Shayman, "Periodic equilibria for the matrix Riccati equation with periodic coefficients", Report No. SSM 8309, Dept. of Systems Science and Mathematics, Washington University, St. Louis, MO 63130, Oct. 1983.
15. J. C. Willems, "Least squares stationary optimal control and the algebraic Riccati equation", IEEE Trans. Automat. Contr., 16 (1971), 621-634.

ROBUST MINIMAX LOCATION PARAMETER ESTIMATION UNDER UNIFORM LOSS WITH A COMPACT PARAMETER SPACE

MEHMET ZEYTINOGLU & MAX MINTZ
Department of Systems Engineering
University of Pennsylvania
Philadelphia, PA

INTRODUCTION

The standard statement of a minimax location parameter estimation problem includes as given: a parameter space Ω; a space of actions A; a loss function L defined on $A \times \Omega$; and a sampling CDF F( ). If the underlying sampling CDF is imprecisely known, then this standard minimax decision model must be reformulated to account for this additional uncertainty. Statistical decision rules which are applicable in this more general problem setting are referred to as robust procedures.

In this paper we consider robust minimax location parameter estimates under uniform loss with a compact parameter space. We delineate sufficient conditions for the existence of robust and median-robust minimax estimates, as well as corresponding jointly least favorable and jointly median-least favorable prior distributions. We also examine the asymptotic behavior of the median-robust minimax estimates, and describe the extension of this class of rules to the α-robust minimax rules, which are based on the α-trimmed mean of which the median is a special case. The present paper extends our earlier minimax location parameter estimation results, which appear in [1, 2], by addressing the issues of robustness and N-sample decision problems.

PROBLEM STATEMENT

We consider the following robust and median-robust minimax location parameter estimation problems: Let $\underline{Z}$ denote a vector of N IID observations of a scalar random variable Z with CDF $F(z-\theta)$. Assume that F is an unknown element of a given class F, and that θ is an unknown element of the given compact interval $\Omega = [-d, d]$. Let A denote the action space of the Statistician. Here, $A = [-d, d]$. Let $L(a,\theta)$ denote the uniform loss function defined on $A \times \Omega$:

$$L(a,\theta) = 0, \qquad |a-\theta| \leq e;$$

$$L(a,\theta) = 1, \qquad |a-\theta| > e;$$

where $e > 0$, is given. Let $r(\delta,\theta,F) = E[L(\delta,\theta)|\theta,F]$ denote the conditional risk of the decision rule δ given $(\theta,F) \in \Omega \times F$. Based on these previous definitions and assumptions, we wish to determine robust and median-robust minimax estimates for θ.

Definition 1: An estimate $\delta^*(\underline{Z})$ is said to be a robust minimax estimate for θ, if for all $\delta(\underline{Z})$:

$$\sup_{\substack{\theta\in\Omega \\ F\in F}} r(\delta^*,\theta,F) \leq \sup_{\substack{\theta\in\Omega \\ F\in F}} r(\delta,\theta,F).$$

Definition 2: An estimate $\delta^*(Z_M)$ is said to be a median-robust (M-robust) minimax estimate for θ, if for all $\delta(Z_M)$:

$$\sup_{\substack{\theta\varepsilon\Omega \\ F\varepsilon\mathcal{F}}} r(\delta^*,\theta,F) \leq \sup_{\substack{\theta\varepsilon\Omega \\ F\varepsilon\mathcal{F}}} r(\delta,\theta,F),$$

where Z_M denotes the sample median of the N IID observations.

Remarks: In the limiting case when the uncertainty class $\mathcal{F}$ contains exactly one CDF, the problems of robust and M-robust minimax estimation reduce, respectively, to minimax and M-minimax estimation. Definition 2 is motivated by the desire to obtain a degree of data compression -- in those instances where a one-dimensional sufficient statistic for θ fails to exist. Other approaches for achieving a desired degree of data compression include other applications of the α-trimmed mean, or a linear combination of order statistics.

The usual definition of a least favorable prior distribution admits direct generalizations in the context of both robust and M-robust minimax estimation. Let $\mathcal{D}$ denote the set of all real-valued Borel functions of $\underline{Z}$. Let $\mathcal{D}_M$ denote the set of all real-valued Borel functions of Z_M. Let Λ (Λ_M) denote the set of all probability measures λ on $\Omega \times \mathcal{F}$ such that for all $\delta \varepsilon \mathcal{D}$ ($\mathcal{D}_M$):

$$R(\delta,\lambda) = \int_{\Omega \times \mathcal{F}} r(\delta,\theta,F)d\lambda$$

exists.

Definition 3: A probability measure $\lambda^* \varepsilon \Lambda$ is said to be a jointly least favorable prior distribution for (θ,F), if for all $\lambda \varepsilon \Lambda$:

$$\inf_{\delta\varepsilon\mathcal{D}} R(\delta,\lambda^*) \geq \inf_{\delta\varepsilon\mathcal{D}} R(\delta,\lambda).$$

Definition 4: A probability measure $\lambda^* \varepsilon \Lambda_M$ is said to be a jointly median-least favorable (M-least favorable) prior distribution for (θ,F), if for all $\lambda \varepsilon \Lambda_M$:

$$\inf_{\delta\varepsilon\mathcal{D}_M} R(\delta,\lambda^*) \geq \inf_{\delta\varepsilon\mathcal{D}_M} R(\delta,\lambda).$$

Remark: In the limiting case when the uncertainty class $\mathcal{F}$ contains exactly one CDF, the definitions of a jointly least favorable prior distribution and a jointly M-least favorable prior distribution reduce to the usual definitions.

In this paper we consider several general structures for $\mathcal{F}$. In all cases it is assumed that each $F \varepsilon \mathcal{F}$ is symmetric about zero, i.e., $F(x^+) + F(-(x^+)) = 1$, when $x \geq 0$. The notation x^+ and x^- denote respectively the right- and left-hand limits.

Definition 5: $\mathcal{F}$ is a type I uncertainty class with an upper-envelope CDF F_u, if $\mathcal{F} = \{F\colon F(x^-) \leq F_u(x),\ x \leq 0\}$, where F_u has a density function which is unimodal and symmetric about zero.

Definition 6: $\mathcal{F}$ is a type II uncertainty class with upper- and lower-envelope CDF's F_u and F_l, if $\mathcal{F} = \{F\colon F_l(x^-) \leq F(x^-) \leq F_u(x),\ x \leq 0\}$, where F_u has a density function which is unimodal and symmetric about zero.

Remark: If we allow F_u to be substochastic, i.e., F_u has less than unit probability mass in $(-\infty, \infty)$, then the usual ε-contamination models can be represented by type II uncertainty classes.

Definition 7: $\mathcal{F}$ is a type III (nuisance-parameter) uncertainty class with an upper-envelope CDF $F_u = \Phi(\sigma_2)$, if $\mathcal{F} = \{F: F = \Phi(\sigma),\ \sigma \in [\sigma_1, \sigma_2]\}$, where $\Phi(\sigma,x) \leq \Phi(\sigma_2,x)$ when $\sigma \leq \sigma_2$ and $x \leq 0$; and $\Phi(\sigma_2)$ has a density function which is unimodal and symmetric about zero.

Remark: An example of a type III uncertainty class is the set of all zero mean normal distributions with standard deviation $\sigma \in [\sigma_1, \sigma_2]$.

SADDLE-POINT RESULTS

In this section we delineate sufficient conditions for the existence of robust and M-robust minimax estimates, as well as corresponding jointly least favorable and jointly M-least favorable prior distributions. In the sequel we shall refer to a bound $B(d/e,N,F_u)$ which depends on the ratio d/e, the sample size N, and the upper-envelope CDF F_u. The CDF F_u has a density function which is unimodal and symmetric about zero. The uniform loss function L will be parameterized in terms of d/e and e. To avoid trivial cases, we will assume $d/e > 1$.

Lemma 1: If $N = 1$, and $\mathcal{F}$ contains exactly one distribution F_u, then there exists a minimax rule δ^* and a least favorable prior distribution λ^*.

Lemma 2: If $N > 1$, and $\mathcal{F}$ contains exactly one distribution F_u, then there exists a M-minimax rule δ^* and a M-least favorable prior distribution λ^*.

Lemma 3: If $N > 1$, and $\mathcal{F}$ contains exactly one distribution $F_u = N(0,\sigma_2^2)$, then there exists a minimax rule δ^* and a least favorable prior distribution λ^*. Further, δ^* depends on $\underline{Z}$ through the sample mean $\bar{Z}$.

Let $\mathcal{F}$ denote an uncertainty class of type I-III. We employ the following embedding to extend the measure λ^* (of Lemmas 1-3) from Ω to $\Omega \times \mathcal{F}$. We define the extension λ_E^* on $\Omega \times \mathcal{F}$, as the Ω-consistent measure which concentrates its mass on the cylinder set $\Omega \times \{F_u\}$.

Theorem 1: Let δ^* and λ^* denote the minimax rule and least favorable prior distribution obtained in Lemma 1, based on the sample size $N = 1$ and CDF F_u; let $\mathcal{F}$ denote a type I-III uncertainty class with upper-envelope CDF F_u; let λ_E^* denote the extension of λ^* to $\Omega \times \mathcal{F}$. When $N = 1$, there exists a bound $B(d/e,N{=}1,F_u)$, such that if $e \geq B$, then δ^* is a robust minimax rule and λ_E^* is a jointly least favorable prior distribution.

Theorem 2: Let δ^* and λ^* denote the M-minimax rule and M-least favorable prior distribution obtained in Lemma 2, based on the sample size $N > 1$ and CDF F_u; let $\mathcal{F}$ denote a type I-III uncertainty class with upper-envelope CDF F_u; let λ_E^* denote the extension of λ^* to $\Omega \times \mathcal{F}$. There exists a bound $B(d/e,N,F_u)$, such that if $e \geq B$, then δ^* is a M-robust minimax rule and λ_E^* is a jointly M-least favorable prior distribution.

Theorem 3: Let δ^* and λ^* denote the minimax rule and least favorable prior distribution obtained in Lemma 3, based on the sample size $N > 1$ and CDF $F_u = N(0,\sigma_2^2)$; let $\mathcal{F} = \{F: F = N(0,\sigma^2),\ \sigma_1 \leq \sigma \leq \sigma_2\}$; let λ_E^* denote the extension of λ^* to $\Omega \times F$. There exists a bound $B(d/e, N, F_u)$, such that if $e \geq B$, then δ^* is a robust minimax rule and λ^*_E is a jointly least favorable prior distribution.

A proof of Lemma 1 appears in [1,2]. The proofs of Lemmas 2-3, and Theorems 1-3 appear in an extended version [3] of this present paper.

In the remainder of this paper we: (i) review the salient features of the class of minimax rules and corresponding least favorable prior distributions which were obtained in [1,2]; (ii) use the results of Lemma 1 to motivate Lemmas 2-3, and Theorems 1-3; and (iii) provide examples which illustrate various applications of these lemmas and theorems.

DISCUSSION

In this section we examine the relationships between Lemma 1 and the new results embodied in Lemmas 2-3, and Theorems 1-3. We consider the case when d is an integer multiple of e. The analyses of the noninteger cases appear in [1-3].

Lemma 1: Our "constructive" proof of Lemma 1 [1,2] provides an explicit structural determination of the minimax rules and corresponding least favorable prior distributions for the given class of location parameter estimation problems characterized by a uniform loss function and a compact interval parameter space. Here, $N = 1$ and F is a known CDF with a density function which is unimodal and symmetric about zero. Let $d = (2n+1)e + c$, where $n = 1,2,\ldots,$ and the parameter c equals zero (e) if d is an odd (even) multiple of e. Our construction of δ^* is based on a sequence of observations which we develop in this section. Due to the existing symmetry in this problem setting, function definitions will be stated for nonnegative arguments.

Let Δ denote the following parameterized family of decision rules δ:

$$\delta(Z) = \begin{cases} d-e, & c+a_n+2ne \leq Z; \\ \cdot & \\ \cdot & \\ Z-a_2, & c+a_2+2e \leq Z < c+a_2+4e; \\ 2e+c, & c+a_1+2e \leq Z < c+a_2+2e; \\ Z-a_1, & c+a_1 \leq Z < c+a_1+2e; \\ c, & c \leq Z < c+a_1; \\ Z, & 0 \leq Z < c; \end{cases} \tag{1}$$

where: $0 \leq a_1 \leq a_2 \leq \cdot\cdot\cdot \leq a_n < \infty$. The selection of the parameters $a_1, a_2, \ldots, a_n$ will be described in the sequel.

Let M denote the following parameterized family of probability distributions λ, with density functions $f(\theta)$, defined on Ω:

$$f(\theta) = \sum_{i=1}^{n+1} k_i[U(\theta+(2i-1)e+c) - U(\theta-(2i-1)e-c)]; \tag{2}$$

where: $U(t) = 1$, if $t \geq 0$; $U(t) = 0$, if $t < 0$; and the parameters $\{k_i\}$, are nonnegative normalizing constants.

Observation 1. Each $\delta \in \Delta$ (1) is Bayes with respect to an a priori density function, $f \in M$, for some choice of parameters $\{k_i\}$.

Observation 2. The conditional risk function of each $\delta \varepsilon \Delta$ is given by:

$$r(\delta,\theta,F) = \begin{cases} F(a_n-e), & d-2e < \theta \leq d; \\ F(a_{n-1}-e), & \theta = d-2e; \\ F(-a_n-e)+F(a_{n-1}-e), & d-4e < \theta < d-2e; \\ \cdot & \\ \cdot & \\ \cdot & \\ F(-a_2-e)+F(a_1-e), & c+e < \theta < c+3e; \\ F(-a_2-e)+(c/e)F(-e)+(1-c/e)F(-a_1-e), & \theta = c+e; \\ F(-a_1-e)+(c/e)F(-e)+(1-c/e)F(-a_1-e), & 0 < \theta < c+e; \\ 2F(-a_1-e), & \theta = 0. \end{cases} \tag{3}$$

We observe that $r(\delta,\theta,F)$ is piecewise constant over the sets of a finite partition of Ω. We note specifically, that $r(\delta,\theta,F)$ is piecewise constant over n+1 nondegenerate subintervals of [0, d], where a nondegenerate interval is an interval which consists of more than a single point.

Based on this latter observation, we define an (n+1)-dimensional vector $J(\underline{a})$, with components equal to $r(\delta,\theta,F)$, over the n+1 nondegenerate partition intervals of [0, d].

$$\begin{aligned} J(\underline{a}) = [&(F(-a_1-e)+(c/e)F(-e)+(1-c/e)F(-a_1-e)), \\ &(F(-a_2-e)+F(a_1-e)), \\ &\cdot \\ &\cdot \\ &(F(a_n-e))]^T, \end{aligned} \tag{4}$$

where: $\underline{a} = (a_1, a_2, .., a_n)^T \varepsilon R \subset E^n$, such that $0 \leq a_1 \leq a_2 \leq . . . \leq a_n < \infty$.

Observation 3. For each $\delta \varepsilon \Delta$, the max $r(\delta,\theta,F)$ occurs at one or more of the nondegenerate subintervals in (3).

Observation 4. The parameter vector $\underline{a} \varepsilon R$, can be chosen to equalize the components of $J(\underline{a})$ to a common value $\overline{M}$, where $M < 2F(-e)$.

Observation 5. If $\underline{a} \varepsilon R$ satisfies the equalization condition of Observation 4, then δ^* is minimax.

This latter observation follows by noting that δ^* is a Bayes rule with respect to a prior distribution λ^*, by Observation 1; and that λ^* assigns probability one to the subset of Ω on which $r(\delta^*,\theta,F)$ attains its maximum value, by Observations 3 and 4.

Observation 6. The prior distribution λ^* referred to in Observation 5 is a least favorable prior distribution.

Observation 7. Let F' and F'' denote CDF's which individually satisfy the hypothesis of Lemma 1. Let r' and r'' denote, respectively, the minimax risk based on F' and F''. If $F'(x) < F''(x)$ $(F'(x) \leq F''(x))$ for all $x < 0$, and $e \geq B(d/e,N=1,F'')$ -- a given bound, then $r' < r''$ $(r' \leq r'')$.

Example 1 [1,2]: Let $N = 1$, $d = 3e$, $e = 0.1$, and $F = N(0,1)$. By referring to (1) and (3), the minimax rule δ^* and conditional risk function $r(\delta^*,\theta,F)$, are respectively:

$$\delta^*(Z) = \begin{cases} 2e, & a+2e \leq Z; \\ Z-a, & a \leq Z < a+2e; \\ 0, & 0 \leq Z < a; \end{cases}$$

$$r(\delta^*,\theta,F) = \begin{cases} F(a-e), & e < \theta \leq 3e; \\ F(-a-e), & \theta = e; \\ 2F(-a-e), & 0 \leq \theta < e; \end{cases}$$

for some $a > 0$.

Let $J(a) = (2F(-a-e), F(a-e))^T$. Upon equating the two components of $J(a)$, one obtains the relation $2F(-a-e) = F(a-e)$. Solving this last relation for a, we obtain: $a^\circ = 0.4$. The corresponding minimax risk is 0.62.

The corresponding least favorable prior distribution λ^*, with density f^*, is:

$$f^*(\theta) = 0.14[U(\theta+e)-U(\theta-e)] + 1.62[U(\theta+3e)-U(\theta-3e)].$$

Lemma 2: We utilize this lemma when $N > 1$, $\mathcal{F}$ contains exactly one CDF F, and F possesses a density function which is non-Gaussian, unimodal, and symmetric about zero. The following sequence of observations motivates the selection of a M-minimax rule in these circumstances, and illuminates the relationship between Lemmas 1 and 2.

Observation 8. The non-Gaussian symmetric structure of F effectively eliminates the existence of a one-dimensional sufficient statistic for θ.

In order to "apply" Lemma 1 when $N > 1$, we first select a real-valued statistic $T(\underline{Z})$ that possesses a density function which is unimodal and symmetric about θ; we then apply Lemma 1 based on the CDF of $T(\underline{Z}) - \theta$. Examples of candidate T statistics include: the sample mean, the sample median, and other symmetric linear combinations of order statistics. The choice of the sample median in Lemma 2, as the statistic by which we seek to reduce our initial uncertainty in θ, is motivated by the median's well-known consistency property, as well as the relative ease with which the CDF of T can be computed. The value of this reduction of initial uncertainty in θ is motivated by Observation 7. We remark that the sample median has no global optimality property with respect to F. Thus, for a given CDF F, other choices of T may exist which improve on the sample median.

Example 2: Let $N = 3$, $d = 3e$, and $e = 0.2133$. Let F denote the CDF of the double-exponential distribution with unit variance:

$$F(x) = \begin{cases} 1 - (1/2)\mathrm{Exp}(-x), & x \geq 0; \\ (1/2)\mathrm{Exp}(x), & x < 0. \end{cases}$$

Let F' denote the CDF of the centered sample median $Z_M - \theta = T(\underline{Z}) - \theta$. In this example:

$$F'(t) = F^2(t)(3 - 2F(t)).$$

Since $d = 3e$, the M-minimax rule δ^* has the same general structure as the minimax rule δ^* in Example 1. The argument Z is merely replaced by T. By referring to (1) and (3), the M-minimax rule δ^* and conditional risk function $r(\delta^*,\theta,F)$, are respectively:

$$\delta^*(T) = \begin{cases} 2e, & a+2e \leq T; \\ T-a, & a \leq T < a+2e; \\ 0, & 0 \leq T < a; \end{cases}$$

$$r(\delta^*,\theta,F) = \begin{cases} F'(a-e), & e < \theta \leq 3e; \\ F'(-a-e), & \theta = e; \\ 2F'(-a-e), & 0 \leq \theta < e; \end{cases}$$

for some $a > 0$.

Let $J(a) = (2F'(-a-e), F'(a-e))^T$. Upon equating the two components of $J(a)$, one obtains the relation $2F'(-a-e) = F'(a-e)$. Solving this last relation for a, we obtain: $a^\circ = e = 0.2133$. The corresponding M-minimax risk is 0.500.

The corresponding M-least favorable prior distribution λ^*, with density f^*, is:

$$f^*(\theta) = 0.46[U(\theta+e)-U(\theta-e)] + 0.63[U(\theta+3e)-U(\theta-3e)].$$

Lemma 3: We utilize this lemma when $N > 1$, $\mathcal{F}$ contains exactly one CDF F, and $F = N(0,\sigma^2)$. The following sequence of observations provides motivation for the existence of an unrestricted minimax rule in these circumstances, and illuminates the relationships between Lemmas 1-3.

Observation 9. Since the sample mean $\bar{Z} = T(\underline{Z})$ is a sufficient statistic for θ, any minimax rule depends on $\underline{Z}$ through T.

Observation 10. Let F' denote the CDF of the centered sample mean $\bar{Z} - \theta = T(\underline{Z}) - \theta$. Thus, $F' = N(0,\sigma^2/N)$. The desired minimax rule δ^* is obtained by applying Lemma 1 based on the CDF F'.

Observation 11. Lemma 3 replaces Lemma 2 when F is Gaussian. Since the sample mean is a sufficient statistic for θ, the application of Lemma 1, based on F', provides an unrestricted minimax rule and least favorable prior distribution.

Example 3: Let $d = 3e$, $e = 0.1$, and $F = N(0,N)$. Since $F' = N(0,1)$, $d = 3e$, and $e = 0.1$, the minimax rule δ^* is obtained from Example 1, by merely replacing Z by T. The corresponding conditional risk function and least favorable prior distribution are identical to those in Example 1.

Theorem 1: We utilize this theorem when $N = 1$, $\mathcal{F}$ denotes a type I-III uncertainty class with upper-envelope CDF F_u, and $e \geq B(d/e,N=1,F_u)$ -- a given bound. The following sequence of observations provides motivation for the existence of a robust minimax rule in these circumstances, and illuminates the relationship between Theorem 1 and Lemma 1. For brevity, we restrict our attention to the case when $d = 3e$. Let δ^* denote the minimax rule obtained by applying Lemma 1, based on the CDF $F = F_u$. Let $a°$ denote the solution to: $2F_u(-a-e) = F_u(a-e)$.

Observation 12. Based on this application of Lemma 1, the minimax rule δ^* and corresponding conditional risk function $r(\delta^*,\theta,F_u)$ are, respectively:

$$\delta^*(Z) = \begin{cases} 2e, & a+2e \leq Z; \\ Z-a, & a \leq Z < a+2e; \\ 0, & 0 \leq Z < a; \end{cases}$$

$$r(\delta^*,\theta,F_u) = \begin{cases} F_u(a-e), & e < \theta \leq 3e; \\ F_u(-a-e), & \theta = e; \\ 2F_u(-a-e), & 0 \leq \theta < e; \end{cases}$$

for $a = a°$.

Observation 13. If $e \geq -(1/2)F_u^{-1}(1/4)$, then $a° \leq e$.

Observation 14. If $e \geq -(1/2)F_u^{-1}(1/4)$, then $r(\delta^*,\theta,F) \leq r(\delta^*,\theta,F_u)$ for all $F \in \mathcal{F}$. Hence, δ^* is a robust minimax rule.

Example 4 (an ε-contaminated normal distribution): Let $d = 3e$, and $\mathcal{F} = \{F\colon F = (1-\varepsilon)\Phi + \varepsilon H\}$, where: $\Phi = N(0,1)$, H is any CDF which is symmetric about zero, and $0 < \varepsilon < 1/2$. The corresponding upper-envelope CDF is $F_u = (1-\varepsilon)\Phi + \varepsilon/2$. Here, F_u is substochastic. Let $B = -(1/2)F_u^{-1}(1/4)$. Thus, $B = -(1/2)\Phi^{-1}[(1-2\varepsilon)/4(1-\varepsilon)]$. Let $a°$ denote the unique solution to: $2F_u(-a-e) = F_u(a-e)$, or equivalently: $2\Phi(-a-e) = \Phi(a-e) - (\varepsilon/2)/(1-\varepsilon)$. Thus, if $e \geq B$, then δ^* is a robust minimax rule with corresponding parameter value $a°$.

Theorem 2: We utilize this theorem when $N > 1$, $\mathcal{F}$ denotes a type I-III uncertainty class with upper-envelope CDF F_u, and $e \geq B(d/e,N,F_u)$ -- a given bound. The following sequence of observations provides motivation for the existence of a M-robust minimax rule in these circumstances, and illuminates the relationship between Theorem 2 and Lemma 2. For brevity, we restrict our attention to the case when $d = 3e$. Let δ^* denote the M-minimax rule obtained by applying Lemma 2, based on the CDF $F = F_u$. Let $a°$ denote the solution to: $2F'(-a-e) = F'(a-e)$, where F' denotes the CDF of the centered sample median $Z_M - \theta = T(\underline{Z}) - \theta$.

Observation 15. Based on this application of Lemma 2, the M-minimax rule δ^* and corresponding conditional risk function $r(\delta^*,\theta,F_u)$ are, respectively:

$$\delta^*(T) = \begin{cases} 2e, & a+2e \leq T; \\ T-a, & a \leq T < a+2e; \\ 0, & 0 \leq T < a; \end{cases}$$

$$r(\delta^*,\theta,F_u) = \begin{cases} F'(a-e), & e < \theta \leq 3e; \\ F'(-a-e), & \theta = e; \\ 2F'(-a-e), & 0 \leq \theta < e; \end{cases}$$

for $a = a°$.

Observation 16. If $e \geq B(d/e,N,F_u) = -(1/2)F'^{-1}(1/4)$, then $a° \leq e$.

Observation 17. If G' denotes the CDF of the centered sample median based on $G \in \mathcal{F}$, then $G'(x^-) \leq F'(x)$, when $x < 0$.

Observation 18. If $e \geq -(1/2)F'^{-1}(1/4)$, then $r(\delta^*,\theta,G) \leq r(\delta^*,\theta,F_u)$ for all $G \in \mathcal{F}$. Hence, δ^* is a M-robust minimax rule.

Observation 19. Since the median is a consistent estimate of θ, the bound $B(d/e,N,F_u) = -(1/2)F'^{-1}(1/4)$ tends to zero with increasing sample size N. Hence, there exists an integer $N'(d/e,e,F_u)$ such that if $N \geq N'$, then $e \geq B$. Thus, for each $N \geq N'$ there exists a M-robust minimax rule.

Example 5 (Example 2 revisited and extended): Let $N = 3$, $d = 3e$, and $e = 0.2133$. Let $\mathcal{F}$ denote a type I uncertainty class with upper-envelope CDF F_u, where F_u denotes the CDF of the double-exponential distribution with unit variance. Let F' denote the CDF of the centered sample median based on F_u. Let δ^* denote the M-minimax rule obtained in Example 2. Since $e = -(1/2)F'^{-1}(1/4)$, δ^* is a M-robust minimax rule for θ. The corresponding M-robust minimax risk is 0.500.

Observation 20. For large N, one may approximate the CDF F' by its asymptotic form: $N(0,1/4Nf^2(0))$, where $f(0)$ denotes $dF_u/dx|x=0$ [4]. Thus, for large N, the computation of M-robust and M-minimax rules is simplified by the existence of a normal approximation to F'.

Theorem 3: We utilize this theorem when $N > 1$, $\mathcal{F} = \{F: F = N(0,\sigma^2), \sigma_1 \leq \sigma \leq \sigma_2\}$, $F_u = N(0,\sigma_2^2)$, and $e \geq B(d/e,N,F_u)$ -- a given bound. The following sequence of observations provides motivation for the existence of a robust minimax rule in these circumstances, and illuminates the relationship between Theorem 3 and Lemma 3. For brevity, we restrict our attention to the case when $d = 3e$. Let δ^* denote the minimax rule obtained by applying Lemma 3, based on the CDF $F = F_u$. Let a° denote the solution to: $2F'(-a-e) = F'(a-e)$, where F' denotes the CDF of the centered sample mean $\bar{Z} - \theta = T(\underline{Z}) - \theta$, and $F' = N(0,\sigma_2^2/N)$.

Observation 21. Based on this application of Lemma 3, the minimax rule and corresponding conditional risk function $r(\delta^*,\theta,F\ )$ are, respectively:

$$\delta^*(T) = \begin{cases} 2e, & a+2e \leq T; \\ T-a, & a \leq T < a+2e; \\ 0, & 0 \leq T < a; \end{cases}$$

$$r(\delta^*,\theta,F_u) = \begin{cases} F'(a-e), & e < \theta \leq 3e; \\ F'(-a-e), & \theta = e; \\ 2F'(-a-e), & 0 \leq \theta < e; \end{cases}$$

for a = a°.

Observation 22. If $e \geq B(d/e,N,F_u) = -(1/2)F'^{-1}(1/4)$, then $a° \leq e$.

Observation 23. If G' denotes the CDF of the centered sample mean based on $G \in \mathcal{F}$, then $G'(x^-) \leq F'(x)$, when $x < 0$.

Observation 24. If $e \geq -(1/2)F'^{-1}(1/4)$, then $r(\delta^*,\theta,G) \leq r(\delta^*,\theta,F_u)$ for all $G \in \mathcal{F}$. Hence, δ^* is a robust minimax rule.

Observation 25. Since the mean is a consistent estimate of θ, the bound $B(d/e,N,F_u) = -(1/2)F'^{-1}(1/4)$ tends to zero with increasing sample size N. Hence, there exists an integer $N'(d/e,e,F_u)$ such that if $N \geq N'$, then $e \geq B$.

Thus, for each $N \geq N'$ there exists a robust minimax rule.

EXTENSIONS

The large sample case permits the generalization of M-robust minimax estimation to α-robust minimax estimation. Here, the statistic $T(\alpha,\underline{Z})$ denotes the α-trimmed mean. We first seek a pair (α^*,F^*), which obtains the:

$$\min_{\alpha} \max_{F} V(\alpha,F),$$

where V denotes the asymptotic variance of the α-trimmed mean for a given CDF F [4,5,6]. Here, $\alpha \in [0,1/2]$, and $F \in \mathcal{F}$. We next approximate the CDF F' of $T(\alpha^*,\underline{Z}) - \theta$ based on this asymptotic analysis, under the assumption $F = F^*$. The α-robust minimax rule is then obtained by applying Lemma 2 based on $F_u = F'$, with scaling to account for the true (finite) sample size N. Applications of this asymptotic technique appear in [3].

REFERENCES

[1] Zeytinoglu, M. and Mintz, M. (1982). Minimax Location Parameter Estimation Under Uniform Loss with a Compact Parameter Space. Proc. of the Twentieth Annual Allerton Conference on Communication, Control, and Computing.

[2] An extended version of [1]. (1982). Submitted for publication.

[3] Zeytinoglu, M. and Mintz, M. (1983). Robust Minimax Location Parameter Estimation Under Uniform Loss with a Compact Parameter Space. (An extended version of this present paper.) Submitted for publication.

[4] Bickel, P.J. (1965). On Some Robust Estimates of Location. Ann. Math. Statist. 36, 847-858.

[5] Gastwirth, J.L. and Rubin, H. (1969). On Robust Linear Estimates. Ann. Math. Statist. 40, 24-39.

[6] Huber, P.J. (1981). Robust Statistics. Wiley, New York.

ACKNOWLEDGMENT

The authors wish to express their gratitude for the following support. The research of the first author was supported in part by a Fulbright Grant. The research of the second author was supported in part by a National Research Council Associateship at the US Army Ballistic Research Laboratory.

ROBUST ESTIMATION AND DETECTION OF SIGNALS WITH ARBITRARY PARAMETERS

PATRICK A. KELLY
Program in Computer, Information, and Control Engineering
The University of Michigan
Ann Arbor, Michigan 48109

ABSTRACT

Minimax-robust location parameter estimation and known signal detection results are extended to problems involving signals with arbitrary vector parameters.

I. INTRODUCTION

In [1], Huber considered the problem of robustly estimating the location of a distribution based on observations drawn from a sequence $\{Z_i = X_i + \theta_o\}$ of iid random variables. He showed that M-estimates (i.e., those based on minimizing $\sum_{i=1}^{n} \rho(Z_i - \theta)$ with respect to θ, where $\rho \epsilon H$, a class of convex functions) are asymptotically normal with variances depending on ρ and the distribution of X_i. When the possible distributions have densities forming a class F, Huber found that in some cases a "least-favorable" density f_o exists such that, with ρ_o defined to be $-\log f_o$, the asymptotic variance of the estimate (denoted $V^2(f, \rho)$) has a saddlepoint at (f_o, ρ_o). That is,

$$\inf_{\rho \epsilon H} V^2(f_o, \rho) = V^2(f_o, \rho_o) = \sup_{f \epsilon F} V^2(f, \rho_o)$$

The estimate derived using ρ_o is then the minimax-robust estimate of location. The amplitude of a known signal added to noise also plays the role of a location parameter, and in [2] El-Salwy and Vandelinde applied Huber's results to the detection of known signals; they showed that, asymptotically, a test based on the robust amplitude estimate satisfies a minimax Neyman-Pearson criterion.

In many detection problems, however, the signal is not known beforehand, but is instead a member of a set of parameterized functions. The task is to decide, based on noisy observations, if a function from the set is present, and, if so, to estimate its parameter values. The method used in [1] depends on $E[\partial \rho(Z - \theta)/\partial \theta]$ being a monotone function of θ;

no such property holds for this case. The signal parameters do affect the observations only by changing their means, however, so one might expect a robust estimator of their values to have the same form as a robust estimator of location. In section II this will be shown to be true. In section III it is shown that, if one of the signal parameters is amplitude, a test based on its robust estimate is still asymptotically minimax. Proofs are sketched.

II. PARAMETER ESTIMATION

The estimation problem to be considered is the following: Given observations $Z_i = X_i + s_i(\beta)$ for $i = 1, 2, \ldots$, where $\beta = \{\beta_1, \ldots, \beta_k\}$ lies in a compact subset Θ of R^k, $\{s_i(\beta)\}$ is the sequence of parameterized signal points, and $\{X_i\}$ is a sequence of iid random variables having a probability density f (satisfying conditions specified later), examine the properties of M-estimates of β; that is, of estimates $\hat{\beta}_n$ defined as the values minimizing $\sum_{i=1}^{n} \rho(Z_i - s_i(\theta))$ over Θ, where ρ is an element of H. Following the pattern of [3], in which asymptotic normality is proved for M-estimates of arbitrary parameters of the distribution of each member of an iid sequence, we will show first that, under certain conditions, $\hat{\beta}_n \to \beta$ a. s., and secondly that $\sqrt{n}\,(\hat{\beta}_n - \beta)$ is an asymptotically Gaussian random vector with a covariance matrix having a saddlepoint property like that of the variance of M-estimates of location.

A. Almost-Sure Convergence

Assume that:

(1) s_i is a continuous function of θ, uniformly in i;

(2) ρ is strictly convex, continuous, and symmetric;

(3) for all finite M, $E\rho^2(X+M) < \infty$;

(4) for all $\theta \in \Theta$ there exists $m(\theta) < \infty$ such that $|s_i(\theta)| < m(\theta)$ for all i;

and (5) for all $\theta \neq \beta$, $\lim_n \frac{1}{n} \sum_{i=1}^{n} E\rho(Z_i - s_i(\theta)) > E\rho(X)$.

Lemma 1. $\hat{\beta}_n \to \beta$ a. s.

Proof: Let U be any open neighborhood of β, and define $h(\theta)$ to be

$\lim_{n} \frac{1}{n} \sum_{i=1}^{n} E\rho(Z_i - s_i(\theta))$. The assumptions imply that h is continuous, so it reaches its inf on $\Theta \setminus U$, and, by assumption (5), there exists $\epsilon > 0$ such that $\inf_{\theta \epsilon \Theta \setminus U} h(\theta) \geq E\rho(X) + 2\epsilon$. In the manner of the proof of Thm. 1 of [3], this and the strong law of large numbers can be shown to imply that, for large enough n,

$$\inf_{\theta \epsilon \Theta \setminus U} \frac{1}{n} \sum_{i=1}^{n} \rho(Z_i - s_i(\theta)) \geq \frac{1}{n} \sum_{i=1}^{n} \rho(Z_i - s_i(\beta)) + \epsilon \quad \text{a. s.}$$

That is, $\hat{\beta}_n \epsilon U$ a. s. for sufficiently large n.

B. Asymptotic Normality

To show asymptotic normality, further definitions and assumptions must be made, corresponding to those in section 4 of [3]. Assume that β is an interior point of Θ, that ρ is continuously differentiable, and that for each i s_i is a differentiable function of θ. Define $\psi(x)$ to be $\frac{d\rho(x)}{dx}$, and let $s_{ij}(\theta)$ denote the j^{th} component of $\nabla s_i(\theta)$. Assume that $E\psi(X) = 0$ (it will if f is symmetric). Then, since $\hat{\beta}_n$ minimizes $\sum_{i=1}^{n} \rho(Z_i - s_i(\theta))$, we have that

$$\sum_{i=1}^{n} \psi(Z_i - s_i(\hat{\beta}_n)) \nabla s_i(\hat{\beta}_n) = 0 \quad \text{a. s.}$$

for n sufficiently large. Define

$$\text{(i)} \quad \lambda(y) = E\psi(X+y);$$

$$\text{(ii)} \quad \lambda_i(\theta) = \lambda(s_i(\beta) - s_i(\theta)) \nabla s_i(\theta);$$

$$\text{and (iii)} \quad U_i(x, \theta, d) = \sup_{\|\tau - \theta\| \leq d} \| \psi(x + s_i(\beta) - s_i(\tau)) \nabla s_i(\tau) - \psi(x + s_i(\beta) - s_i(\theta)) \nabla s_i(\theta) \|;$$

where $\|\cdot\|$ indicates sup norm on R^k. Assume that:

(6) there exists $M_o^2 < \infty$ such that $\|\nabla s_i(\beta)\|^2 E\psi^2(X) < M_o^2$ for all i;

(7) there exist $d_1 > 0$, $a > o$ and N such that $n \geq N$ and $\|\theta - \beta\| < d_1$ imply that $\frac{1}{n} \|\sum_{i=1}^{n} \lambda_i(\theta)\| \geq a \|\theta - \beta\|$;

and (8) there exist $d_2 > 0$, $b > 0$, and $c > 0$ such that for any $0 \leq d \leq d_2$, $\|\theta - \beta\| + d \leq d_2$ implies that $E U_i(X, \theta, d) \leq b \cdot d$ and $E U_i^2(X, \theta, d) \leq c \cdot d$ for all i.

Theorem 1. $\| \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \psi(X_i) \nabla s_i(\beta) + \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \lambda_i(\hat{\beta}_n) \| \xrightarrow{P} 0$.

Proof. The proof is a minor modification of that of Thm. 3 of [3] .

Lemma 2. $\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \psi(X_i) \nabla s_i(\beta) \xrightarrow{D} N(0, E\psi^2(X) \cdot C)$

where C is a $k \times k$ matrix with $(j,\ell)^{th}$ component equal to

$\lim_n \frac{1}{n} \sum_{i=1}^{n} s_{ij}(\beta) s_{i\ell}(\beta)$.

Proof: For any set of real numbers $\{\alpha_1, \ldots, \alpha_k\}$, let τ_i denote the sum $\sum_{j=1}^{k} \alpha_j s_{ij}(\beta)$ and define Y_i to be $\tau_i \psi(X_i)$. By the Central Limit Theorem,

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} Y_i \xrightarrow{D} N(0, E\psi^2(X) \cdot \lim_n \frac{1}{n} \sum_{i=1}^{n} \tau_i^2)$$

But, by Thm 7.7 of [4] , this can be true only if the lemma's statement is true.

Corollary. $\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \lambda_i(\hat{\beta}_n) \xrightarrow{D} N(0, E\psi^2(X) \cdot C)$

Lemma 3. Assume that C is positive definite, and that:

(i) s_{ij} is a continuous function of θ, uniformly in i, for each $j = 1, \ldots, k$;

(ii) In some neighborhood of β, ∇s_{ij} is a continuous function of θ, uniformly in i, for each $j = 1, \ldots, k$;

and (iii) In some neighborhood of the origin, λ is continuously differentiable, with $|\lambda'(0)| > 0$.

Then $\sqrt{n}(\beta_n - \beta) \xrightarrow{D} N(0, (E\psi^2(X)/[\lambda'(0)]^2) \cdot C^{-1})$.

Proof. Let $\lambda_{ij}(\theta)$ denote the j^{th} component of $\lambda_i(\theta)$, and define Λ_n to be the $k \times k$ matrix with $(j,\ell)^{th}$ component $\frac{1}{n} \sum_{i=1}^{n} \frac{\partial \lambda_{ij}(\beta)}{\partial \theta_\ell}$. The assumptions in the lemma imply that $\frac{\partial \lambda_{ij}}{\partial \theta_\ell}$ is a continuous function of θ, uniformly in i, in some neighborhood of β. Then the Mean Value Theorem and the a. s. convergence of $\hat{\beta}_n$ to β can be used to show that, for any $\epsilon > 0$,

$$\| \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \lambda_i(\hat{\beta}_n) - \sqrt{n} \Lambda_n (\hat{\beta}_n - \beta) \| \leq \epsilon \| \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \lambda_i(\hat{\beta}_n) \| \text{ a. s.}$$

for n sufficiently large. Since $\Lambda_n \to -\lambda'(0) C$, this implies that

$$\left\| \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \lambda_i(\beta_n) + \lambda'(0)\, C\sqrt{n}(\beta_n - \beta) \right\| \xrightarrow{P} 0 .$$

Thus, $\lambda'(0)\, C\sqrt{n}(\beta_n - \beta) \xrightarrow{D} N(0, E\psi^2(X) \cdot C)$. If C is positive definite, its inverse exists; the conclusion of the lemma then follows from an application of Thm. 5.1 of [4].

The covariance matrix of the estimate error is in the form of a matrix depending only on the actual signal multiplied by the scalar $E\psi^2(X)/[\lambda'(0)]^2$. This scalar is $V^2(f, \rho)$; that is, it is the variance function found in [1] for location parameter estimation. Thus, the minimax results obtained in [1] apply here, too. In particular, if the possible densities for X_i form a class of symmetric ϵ-contaminated variations of a symmetric and unimodal nominal, then the asymptotic error covariance has a saddlepoint at $(f_o, -\log f_o)$, where f_o is the "least-favorable" density defined in Thm. 1 of [1]. The minimax-robust estimate of the signal parameters is thus formed by setting ρ equal to $\rho_o = -\log f_o$.

III. SIGNAL DETECTION

It will be assumed for this section that the signal may be written in the form $\{b\, s_i(\theta)\}$, where now θ is an interior point of a compact subset Θ of R^{k-1}, the signal is normalized so that $\lim_n \frac{1}{n} \sum_{i=1}^{n} s_i^2(\theta) = 1$ for all $\theta \in \Theta$, and b is any real non-zero number. We are concerned with the problem of deciding between two hypotheses on the observations $\{Z_i\}$:

$$H_o : Z_i = X_i \qquad i = 1, 2, \ldots$$

$$\text{or} \quad H_1 : Z_i = X_i + b_1 s_i(\theta_1), \quad b_1 \neq 0$$

where b_1 and θ_1 are unknown. Define $\hat{b}_{\rho n}$, $\hat{\theta}_{\rho n}$ to be the M-estimates of b_1 and θ_1 (i.e., to be the values minimizing $\sum_{i=1}^{n} \rho(Z_i - b\, s_i(\theta))$ over $R \times \Theta$). Let all of the assumptions on the noise distributions and estimator functions made in section II continue to hold, and let the signal assumptions continue to hold under H_1. Then slight changes in the procedures of section II can be made to show that, under H_1, $\sqrt{n}(\hat{b}_{\rho n} - b_1) \xrightarrow{D} N(0, V^2(f, \rho))$ while under H_0, $\sqrt{n}\, \hat{b}_{\rho n} \xrightarrow{D} N(0, V^2(f, \rho))$.

Consider a test of the form

$$\phi_{\rho n}(Z_1, \ldots, Z_n) = 1 \text{ if } |\hat{b}_{\rho n}| \geq T_n$$
$$= 0 \text{ otherwise}$$

where T_n is some threshold. We decide H_1 to be true if $\phi_{\rho n} = 1$. Then the probability of false alarm with this test if the noise density is f is denoted $P_{FA}(\phi_{\rho n}|f)$ and is asymptotically equal to $2[1 - \Phi(\sqrt{n}\, T_n / V(f, \rho))]$. Similarly, the probability of detection is denoted $P_{DET}(\phi_{\rho n}|f, b_1)$ and, assuming that $T_n \geq O(\frac{1}{\sqrt{n}})$, is asymptotically equal to $\Phi(\sqrt{n}(|b_1| - T_n)/V(f, \rho))$.

Lemma 4. Let F be a class of densities for which the saddlepoint property of $V^2(f, \rho)$ holds, and let T_n be chosen equal to $\frac{1}{\sqrt{n}} V(f_o, \rho_o)\, \Phi^{-1}(1 - \frac{\alpha}{2})$. Then asymptotically:

$$\text{(i)} \quad P_{FA}(\phi_{\rho_o n}|f) \leq P_{FA}(\phi_{\rho_o n}|f_o) = \alpha \text{ for all } f \in F;$$

$$\text{and} \quad \text{(ii)} \quad \text{if } |b_1| > T_n, \; P_{DET}(\phi_{\rho_o n}|f, b_1) \geq P_{DET}(\phi_{\rho_o n}|f_o, b_1)$$

$$\geq P_{DET}(\phi_{\rho n}|f_o, b_1) \text{ for all } f \in F \text{ and for all } \rho \in H.$$

Proof: The proof follows from the saddlepoint property of $V^2(f, \rho)$ and is the same as that of Lemma 2 of [2].

Now, suppose that θ_1 is known; consider the test

$$\phi'_{\rho_o n}(Z_1, \ldots, Z_n) = 1 \text{ if } |\hat{b}'_{\rho_o n}| \geq T_n$$
$$= 0 \quad \text{otherwise}$$

where $\hat{b}'_{\rho_o n}$ minimizes $\sum_{i=1}^{n} \rho_o(Z_i - b\, s_i(\theta_1))$ over R and T_n is as in Lemma 4. Suppose further that F is such that f_o satisfies assumptions 1, 2, and 3 of [5]. (This is true if, for example, F consists of symmetric ϵ-contaminations of a Gaussian nominal). Then Thm. 3 of [5] may be modified to show that, if f_o is the true density, $\phi'_{\rho_o n}$ is an asymptotically most powerful unbiased test between H_o and H_1 of level α. The asymptotic variances of $\hat{b}'_{\rho_o n}$ and $\hat{b}_{\rho_o n}$ are the same, so the probabilities of false alarm and detection under $\phi'_{\rho_o n}$ and $\phi_{\rho_o n}$ are asymptotically the same, implying that, under f_o, the test $\phi_{\rho_o n}$ is an asymptotically most powerful unbiased test between H_o and H_1 of level α when θ_1 is unknown. We thus have:

Lemma 5. If f_o satisfies assumptions 1, 2, and 3 of [5], then conclusion (ii) of Lemma 4 may be strengthened to:

> (ii') $P_{DET}(\phi_{\rho_o n}|f_o, b_1) \geq P_{DET}(\phi\,|f_o, b_1)$ for all unbiased tests ϕ of level α and for all $b_1 \neq 0$; and, if $|b_1| > T_n$, $P_{DET}(\phi_{\rho_o n}|f, b_1) \geq P_{DET}(\phi_{\rho_o n}|f_o, b_1)$ for all $f \epsilon F$.

That is, $\phi_{\rho_o n}$ is a minimax-robust test in the sense that, asymptotically, $P_{FA}(\phi_{\rho_o n}|f)$ is maximized by $P_{FA}(\phi_{\rho_o n}|f_o)$, $P_{DET}(\phi|f_o, b_1)$ is maximized for unbiased tests of level α by $P_{DET}(\phi_{\rho_o n}|f_o, b_1)$, and, for $|b_1| > T_n$, $P_{DET}(\phi_{\rho_o n}|f, b_1)$ is minimized by $P_{DET}(\phi_{\rho_o n}|f_o, b_1)$.

IV. CONCLUSION

This paper has shown that location parameter estimation results in [1] and known signal detection results in [2] may be extended to estimation and detection of signals with arbitrary parameters. The robust M-estimator of the parameter values has the same form as the robust M-estimator of location, and the robust unbiased detector consists of comparing the absolute value of the robust estimate of signal amplitude to a threshold.

REFERENCES

[1] P. J. Huber, "Robust Estimation of a Location Parameter," *Ann. Math. Stat.*, Vol. 35, pp. 73-101, March 1964.

[2] A. H. El-Salwy and V. D. Vandelinde, "Robust Detection of Known Signals," *IEEE Trans. Info. Theory*, Vol. IT-23, pp. 722-727, November 1977.

[3] P. J. Huber, "The Behavior of Maximum Likelihood Estimates under Nonstandard Conditions," *Proc. 5th Berkeley Symp. Math. Stat. and Prob.*, pp. 221-233, 1967.

[4] P. Billingsley, *Convergence of Probability Measures*, New York: Wiley, 1968.

[5] A. Wald, "Asymptotically Most Powerful Tests of Statistical Hypotheses," Ann. Math. Stat., Vol. 12, pp. 1-19, March 1941.

ACKNOWLEDGEMENT

This research was supported by the U.S. Nuclear Regulatory Commission under Contract NRC FIN No. A4047 and by the National Science Foundation under Grant ECS-8005960.

FAULT TOLERANT PARALLEL IMPLEMENTATIONS OF THE KALMAN FILTER

Gerard G. L. Meyer and Howard L. Weinert
Electrical Engineering and Computer Science Department
The Johns Hopkins University
Baltimore, Maryland 21218

SUMMARY

Recent technological developments have made parallel processing a viable option for achieving desired computational speed. Given a problem of interest and an associated procedure to obtain its solution, a network of computing elements may be used to produce the solution, provided that an appropriate decomposition of the procedure can be found. For real-time processing problems where the speed constraints may be quite severe, the number of computing elements required can be large. Besides the difficulty associated with the decomposition of the solution procedure, the use of a large number of computing elements introduces a new set of problems that cannot be ignored. The probability that all computing elements produce correct results becomes vanishingly small as the number of computing elements increases. Thus, in the absence of fault detection and recovery schemes, the processing network may produce a result in the required time, but the probability that the result is also the solution of the problem of interest decreases with the number of computing elements in the network. In view of the preceding discussion, it is clear that the issue of speed cannot be separated from the reliability issue.

The standard approach to the speed/reliability issue consists in designing a processing network to achieve the desired speed, and then replicating the network a sufficient number of times to achieve the desired fault detection and recovery capability. This approach is appealing because, once a processing network that meets the computational speed specification has been designed, the replication

does not involve any additional design effort. Thus the part of the effort devoted to achieving a certain reliability is independent of the specific solution procedure chosen. Unfortunately, the standard approach suffers from a major drawback: it results in a large amount of hardware with the associated problems of cost, size, weight, power, cooling, etc. Even if these problems can be overcome, a more fundamental difficulty remains: as the amount of replication increases, the assumptions used to determine the amount of replication are less likely to be correct.

Our approach to the speed/reliability issue consists in first partitioning the solution procedure to meet the reliability requirements, and then in refining the partition to meet the computational speed requirements. The reliability requirements can be met by partitioning only when the problem to be solved exhibits the appropriate properties: the input data to the solution procedure must exhibit some inherent redundancy. For example, if the input data is a sequence of correlated random variables, then each element of the sequence contains information about every other element of the sequence. The initial partitioning is based on the exploitation of this redundancy to achieve the reliability goals. In the second phase of the design effort, the initial partition is refined to meet the computational speed constraints. Our approach is more limited in scope than the standard approach in that it only applies to a limited class of problems. However, this class of problems includes some very important ones; for example, linear least-squares filtering, smoothing and prediction. The main advantage of our approach is that it results in a substantially smaller amount of hardware than the standard approach.

In order to demonstrate our approach and to compare it with the standard one, we will apply them both to the Kalman filtering problem.

ACKNOWLEDGEMENT

This work was supported by the Office of Naval Research under Contract N00014-81-K-0813.

REFERENCES

AVI76 Avizienis, A., Fault Tolerant Systems, *IEEE Trans. Computers,* Vol. C-25, No. 12, December 1976, pp. 1304-1312.

GRA76 Gray, G.F., and Meyer, J.F., Algebraic Properties of Functions Affecting Optimum Fault Tolerant Realizations, *IEEE Trans. Computers,* Vol. C-25, No. 11, November 1976, pp. 1078-1088.

MEY82a Meyer, G. G. L. and H. L. Weinert, Computational Structures for Parallel Linear Regression, *Proc. 16th Annual Conference on Information Sciences and Systems,* Princeton, New Jersey, March 17-19, 1982.

MEY82b Meyer, G. G. L. and H. L. Weinert, Parallel Algorithms and Computational Structures for Linear Estimation Problems, *1982 Office of Naval Research Workshop on Signal Processing in the Ocean Environment,* Annapolis, Maryland, May 11-14, 1982.

MEY82c Meyer, G. G. L. and H. L. Weinert, Parallel Algorithms and Computational Structures for Linear Estimation Problems, *U.S.C. Workshop on VLSI and Modern Signal Processing,* University of Southern California, Los Angeles, California, November 1-3, 1982.

MEY82d Meyer, G. G. L. and H. L. Weinert, An Approach to Reliable Parallel Data Processing, *Proc. 21st IEEE Conference on Decision and Control,* Orlando, Florida, December 8-10, 1982.

OPTIMAL SEQUENTIAL SMOOTHING WITH UNCERTAIN OBSERVATIONS

HALUK DERIN
Department of Electrical and Computer Engineering
University of Massachusetts
Amherst, MA 01003

MURAT ASKAR
Department of Electrical Engineering
Middle East Technical University
Ankara, Turkey

ABSTRACT

Bayes optimal recursive algorithms that do not require growing memory are obtained for the fixed-interval, fixed-point and fixed-lag smoothing with uncertain observations problems. It is assumed that the signal sequence to be estimated is Markov and that the observations may contain the noise alone or the signal corrupted by noise (not necessarily additive). The uncertainty in the observations is governed by a Markov sequence and the observation noise is an independent sequence. Under these assumptions, recursive algorithms that do not require growing memory are devised for the a posteriori density $f(x_k|Y_N)$, for the three types of smoothing problems. The algorithms also yield a sequential likelihood ratio test type detection scheme as to the presence or absence of the signal at each observation. Bayes fixed-interval smoothing algorithm is applied to a Gauss-Markov example. The simulation results for this example are used to compare the MSE performance of the Bayes smoother with that of the linear smoother. It is observed that Bayes smoother performs considerably better than the linear smoother.

I. INTRODUCTION

The optimum linear estimation with uncertain observations problem has been investigated and solved under various assumptions [1] - [5]. The model used in these studies is that, the signal evolves as a wide-sense Markov sequence, often Gauss-Markov, and the observations consist of noise alone or signal plus noise. Nahi [1] obtained the optimum linear (prediction) estimate assuming that the uncertainty sequence is iid or that it consists of a single r.v. Similar recursive algorithms for the filtering case were presented by Askar, Derin and Yurtseven [2]. Monzingo [3] derived the optimum linear smoothing estimate for an iid uncertainty sequence. Hadidi and Schwartz [4] obtained conditions on the uncertainty sequence for the existance of a recursive optimal linear estimate. Monzingo [5] later obtained linear smoothing algorithms for uncertainty sequences satisfying conditions given in [4].

Under non-Gaussian assumptions or if there is uncertainty in the observations even with Gaussian assumptions, the optimum linear estimate is not the minimum MSE estimate. Studies have been carried out to obtain the minimum MSE estimate using Bayes approach [6] - [10]. In these studies it is assumed that the signal is Markov and that it is corrupted by independent noise (not necessarily additive) and that the uncertainty sequence is Markov. This model is more general than the one used in optimum linear estimation studies.

Ho and Lee [6] obtained Bayes optimal filtering and prediction estimates and Jaffer and Gupta [7], [8] obtained optimal filtering and prediction estimates under Markov uncertainty in the observations. Bayes optimal smoothing results with certain observations for fixed-interval, fixed-point and fixed-lag smoothing were obtained by Askar and Derin [9] - [10]. Recently, Askar and Derin [11] obtained Bayes optimal smoothing algorithms for the fixed-interval, fixed-point and fixed-lag smoothing under uncertain observations. These algorithms, however, require a growing memory, which restricts their applicability considerably. In this paper, Bayes optimal smoothing algorithms that do not require growing memory are presented for the three types of smoothing under uncertain observations problems.

II. PROBLEM STATEMENT

It is assumed that $\{x_k\}$, the signal sequence to be estimated is a Markov sequence with the transition pdf $f(x_k|x_{k-1})$ and initial pdf $f(x_1)$ known. The observation at the kth step is given by

$$y_k = g_k(\gamma_k x_k, v_k) \tag{1}$$

where v_k is the observation noise and γ_k is a 0-1 r.v. representing the uncertainty in the kth observation. $\{\gamma_k\}$ is assumed to be a Markov sequence with transition probabilities $P(\gamma_{k+1}|\gamma_k)$ and initial distribution $P(\gamma_1)$ known. It is also assumed that $\{v_k\}$ is an independent sequence and that the sequences $\{x_k\}$, $\{\gamma_k\}$ and $\{v_k\}$ are mutually independent. g_k is an arbitrary measurable function.

Assuming that the observation set $Y_N = \{y_1, y_2, \ldots, y_N\}$ is available, it is desired to find the a posteriori smoothed density of x_k given the observation set Y_N, namely $f(x_k|Y_N)$, $(1 \leq k \leq N)$ in a recursive way. Depending on the type of smoothing to be done, either N or k or N-k is fixed and recursive relationship is obtained under that condition. The optimal smoothed estimate can then be obtained using the a posteriori smoothed density and the criterion of optimality.

III. RECURSIVE ALGORITHMS FOR BAYES SMOOTHING WITH UNCERTAIN OBSERVATIONS

First we present a result that is used in obtaining the recursive algorithms for the fixed-interval, fixed-point and fixed-lag smoothing with uncertain observations problems.

Theorem 1: For the model described in Section II

$$f(x_k, \gamma_k | x_m, \gamma_m, Y_n) = f(x_k, \gamma_k | x_m, \gamma_m, Y_{m-1}) \text{ for } k < m \leq n \tag{2}$$

This result follows from the Markovian properties of the model.

Fixed-Interval Smoothing

In this case a recursive relationship for $f(x_k|Y_N)$ is needed for fixed N and varying k. The desired smoothing a posteriori density $f(x_k|Y_N)$ is obtained by summing $f(x_k, \gamma_k|Y_N)$ over γ_k. A recursive relationship for the a posteriori distribution $f(x_k, \gamma_k|Y_N)$ is given below:

Theorem 2: For the model described in Section II

$$f(x_k,\gamma_k|Y_N)=f(x_k,\gamma_k|Y_k)\sum_{j=0}^{1}\left\{P(\gamma_{k+1}=j|\gamma_k)\cdot\int\frac{f(x_{k+1}|x_k)\;f(x_{k+1},\gamma_{k+1}=j|Y_N)}{f(x_{k+1},\gamma_{k+1}=j|Y_k)}\,dx_{k+1}\right\} \tag{3}$$

The proof of this result follows from Bayes rule and Theorem 1. Note that it is a backward running algorithm, k running from N-1 to 1.

In addition to the transition probabilities $P(\gamma_{k+1}|\gamma_k)$ and $f(x_{k+1}|x_k)$, the filtered density $f(x_k,\gamma_k|Y_k)$ and the predicted density $f(x_{k+1},\gamma_{k+1}|Y_K)$ are needed in the recursion. These densities are computed and stored during the first (forward) pass over the observations. The smoothing densities are then obtained during the second (backward) pass, as indicated in (3). The filtered and predicted densities $f(x_k,\gamma_k|Y_k)$ and $f(x_{k+1},\gamma_{k+1}|Y_k)$ needed in the recursive smoothing expression were previously obtained by Jaffer and Gupta [8]. A somewhat modified version of recursive relationships for these densities are given below:

$$f(x_k,\gamma_k|Y_k) = \frac{f(y_k|x_k,\gamma_k)\;f(x_k,\gamma_k|Y_{k-1})}{\sum_{i=0}^{1}\int f(y_k|x_k,\gamma_k=i)\;f(x_k,\gamma_k=i|Y_{k-1})\,dx_k} \tag{4}$$

$$f(x_{k+1},\gamma_{k+1}|Y_k) = \sum_{i=0}^{1} P(\gamma_{k+1}|\gamma_k=i)\int f(x_{k+1}|x_k)\;f(x_k,\gamma_k=i|Y_k)\,dx_k \tag{5}$$

These results follow from Bayes rule and Markovian properties of the model.

At each step during the forward pass, 4 densities $f(x_k,\gamma_k=i|Y_k)$ and $f(x_{k+1},\gamma_{k+1}=j|Y_k)$ for i,j = 0,1 have to be stored. So, altogether 4N densities need to be stored. Thus, the memory requirement increases linearly with N and not geometrically as in the algorithm reported in [11]. The need to store 4N continuous pdf's and to process (including integration) a number of these at each step may seem to be a formidable task. But the implementation of the algorithm is clearly manageable if $\{x_k\}$ is a discrete state Markov sequence. Otherwise, a near-optimal strategy can be adopted to implement the algorithm by quantizing the range space of the x_k r.v.'s which will involve the storing and processing of discrete probability distributions. Such a near-optimal strategy was used by Jaffer and Gupta [8] for the optimal filtering algorithm.

If decision on the presence of signal in the observation is of interest, the a posteriori probability that the kth observation contains the signal, given Y_N can be obtained by integrating $f(x_k,\gamma_k|Y_N)$ over x_k.

Fixed-Point Smoothing

In this case a recursive relationship for $f(x_k|Y_N)$ is needed for fixed k and increasing N. The desired smoothing a posteriori density $f(x_k|Y_N)$ is obtained as

$$f(x_k|Y_N) = \sum_{i=0}^{1}\sum_{j=0}^{1}\int f(x_k,x_N,\gamma_k=i,\gamma_N=j|Y_N)\,dx_N \tag{6}$$

A recursive relationship for $f(x_k,x_N,\gamma_k,\gamma_N|Y_N)$ is given below:

Theorem 3: For the model described in Section II

$$f(x_k,x_{N+1},\gamma_k,\gamma_{N+1}|Y_{N+1}) = \frac{f(y_{N+1}|x_{N+1},\gamma_{N+1})}{f(y_{N+1}|Y_N)} \cdot \{ \sum_{m=0}^{1} P(\gamma_{N+1}|\gamma_N=m) [\int f(x_{N+1}|x_N) f(x_k,x_N,\gamma_k,\gamma_N=m|Y_N) dx_N] \} \qquad (7)$$

where

$$f(y_{N+1}|Y_N) = \sum_{i=0}^{1} \int f(y_{N+1}|x_{N+1},\gamma_{N+1}=i) f(x_{N+1},\gamma_{N+1}=i|Y_N) dx_{N+1} \qquad (8)$$

To start the recursion, letting N = k+1

$$\begin{aligned} f(x_k,x_{k+1},\gamma_k,\gamma_{k+1}|Y_k) &= f(x_{k+1},\gamma_{k+1}|x_k,\gamma_k) f(x_k,\gamma_k|Y_k) \\ &= f(x_{k+1}|x_k) P(\gamma_{k+1}|\gamma_k)\ f(x_k,\gamma_k|Y_k) \qquad (9) \end{aligned}$$

can be used.

Note that the recursive expression of (7) involves computation of 4 a posteriori densities corresponding to $\gamma_k,\gamma_{N+1}=0,1$. Memory requirements and computational considerations are similar to those of fixed-interval smoothing algorithm. In case signal presence at step k is of interest, $P(\gamma_k|Y_{N+1})$ can be obtained by integrating and summing $f(x_k,x_{N+1},\gamma_k,\gamma_{N+1}|Y_{N+1})$ over x_k,x_{N+1} and γ_{N+1}.

Fixed-Lag Smoothing

In this case, N-k is fixed and a recursion of $f(x_{k+1}|Y_{N+1})$ or its constituent probabilities is desired for increasing k and N. The desired smoothing a posteriori density $f(x_{k+1}|Y_{N+1})$ can be obtained by integrating and summing $f(x_{k+1},x_{N+1},\gamma_{k+1},\gamma_{N+1}|Y_{N+1})$ over x_{N+1},γ_{k+1} and γ_{N+1}. A recursive relationship for $f(x_{k+1},x_{N+1},\gamma_{N+1},\gamma_{N+1}|Y_{N+1})$ is given below:

Theorem 4: For the model described in Section II, the recursive relationship for $f(x_{k+1},x_{N+1},\gamma_{k+1},\gamma_{N+1}|Y_{N+1})$ is obtained in two steps. First using a fixed-interval smoothing step in reverse, the joint density $f(x_{k+1},x_N,\gamma_{k+1},\gamma_N|Y_N)$ is obtained from $f(x_k,x_N,\gamma_k,\gamma_N|Y_N)$ through the solution of the integral equations

$$f(x_k,x_N,\gamma_k=m,\gamma_N=n|Y_N) = \sum_{i=0}^{1} P(\gamma_{k+1}=i|\gamma_k=m) f(x_k,\gamma_k=m|Y_N) \cdot [\int \frac{f(x_{k+1}|x_k) f(x_{k+1},x_N,\gamma_{k+1}=i,\gamma_N=n|Y_N)}{f(x_{k+1},\gamma_{k+1}=i|Y_k)} dx_{k+1}] \qquad m,n = 0,1 \qquad (10)$$

Second, through the use of the fixed-point smoothing algorithm $f(x_{k+1},x_{N+1},\gamma_{k+1},\gamma_{N+1}|Y_{N+1})$ is obtained in terms of $f(x_{k+1},x_N,\gamma_{k+1},\gamma_N|Y_N)$. The recursion can be started by determining $f(x_k,x_N,\gamma_k,\gamma_N|Y_N)$ for any k, by using the fixed-point algorithm.

The memory requirement and other computational considerations are similar to the other two algorithms except the solution of the integral equation in (10) poses an additional difficulty in the case of fixed-lag smoothing.

IV. GAUSS-MARKOV EXAMPLE

The Bayes fixed-interval smoothing algorithm given in Theorem 2 is applied to a Gauss-Markov model described as

$$\begin{aligned} x_{k+1} &= a_k x_k + u_k \qquad k = 1,2,\ldots,N-1 \qquad |a_k| < 1 \\ y_k &= \gamma_k x_k + v_k \qquad k = 1,2,\ldots,N \end{aligned} \tag{11}$$

where $x_1 \sim N(\mu,\sigma^2)$, $u_k \sim N(0,Q)$ and $v_k \sim N(0,R)$. $\{u_k\}$ and $\{v_k\}$ are independent sequences. $\{u_k\}$, $\{v_k\}$, $\{\gamma_k\}$ and x_1 are mutually independent. $\{\gamma_k\}$ is an iid sequence of 0-1 r.v.'s with $P(\gamma_k=1) = P)\gamma_k=1|\gamma_{k-1}) = p_1$. $\{\gamma_k\}$ is taken as iid, because optimum linear estimate does not exist in recursive form for a Markov sequence [4], whereas the Bayes smoother is valid for a Markov uncertainty sequence as well. In (11), a_k is taken to be -0.8 for all k, $E(x_1) = \mu = 0$ and $\mathrm{Var}(x_1) = \sigma^2 = Q/(1-a^2)$. It follows from these assumptions that $E(x_k) = 0$ and $\mathrm{Var}(x_k) = \sigma^2$ for all k. This particular example is similar to the ones considered by previous researchers [4], [5], [8], [11] to compare linear and Bayes estimators.

For different values of p_1, Q and R, the fixed-interval Bayes smoothing algorithm given in Theorem 2 is applied to the system described above. The MSE for the Bayes smoother is obtained through a 1000 run simulation for data length N = 20, and by taking the a posteriori mean as the Bayes smoothing estimate. The computation and storing of the non-Gaussian conditional pdf's $f(x_k,\gamma_k|Y_{k-1})$, $f(x_k,\gamma_k|Y_k)$ and $f(x_k,\gamma_k|Y_N)$ are done by discretizing the range space of the x_k r.v.'s and by approximating the continuous pdf's by their values at these discrete values. An approximation of this kind was also used in [8] to obtain Bayes filtering estimates. During our experimentations it was observed that an astonishingly small number discrete values yields very good approximations to estimates obtained through a large number of discrete values or through continuous pdf's. For example, MSE values calculated with 21 discrete values are within 1% of those calculated with 51 discrete values. Hence, the range space of x_k r.v.'s are discretized to only 21 values and the discrete values [-5.0, -4.5, -4.0,...,0.0, 0.5,..., 5.0] seemed to be adequate for the Q and R used. After each operation the resulting, effectively discrete, distributions were properly normalized to avoid any round-off error accumulation.

The MSE for the linear smoother is obtained analytically, using the optimum linear smoothing algorithms given by Monzingo [5], also for data length N = 20. To observe the effects of the values of p_1, Q and R on the MSE performance of the Bayes and linear smoothers and on their relative performances, MSE are calculated for the following set of parameter values:

i) $p_1 = 0.6$, $Q = 1.0$ and $R = 0.05$,

ii) $p_1 = 0.8$, $Q = 1.0$ and $R = 0.05$,

iii) $p_1 = 0.8$, $Q = 1.0$ and $R = 0.5$,

iv) $p_1 = 0.8$, $Q = 0.5$ and $R = 0.05$.

The Bayes and linear smoother MSE curves for these parameter values are shown in Figures 1-4. The MSE results for the Bayes and linear filter are also shown in these figures, as they are also obtained while determining the corresponding smoother results.

The observation that can be made from these MSE performance curves are:

i) The MSE performance of the Bayes smoother is significantly better than that of the linear smoother. This is the main point and motivation of this paper.

ii) As expected, the smoother MSE performance is better than the filter MSE performance in both Bayes and linear estimation cases.

iii) The improvement of linear smoother over linear filter is relatively insignificant, a fact which is also pointed out and reasons for which are discussed in [11].

iv) The improvement of the Bayes smoother over the linear smoother is less pronounced for larger observation noise, i.e., larger R.

v) As p_1 approaches unity, the Bayes and linear estimators become identical since the uncertainty is eliminated. This fact was verified in our experimentations but is not demonstrated here.

vi) As expected, the smoother MSE in both Bayes and linear cases, are not as good in the first few and last few steps, since only data in one direction is used in estimating them.

V. CONCLUSION

The Bayesian approach to the estimation problem is appealing because it yields the optimum MSE estimate, whereas the linear estimation approach does not, in many cases of interest. Also, the model used in Bayesian approach is more general than the one used in linear estimation approach. Furthermore, for some cases of interest, such as Markov uncertainty sequence, the optimal linear estimate does not exist in recursive form.

These points constitute the major motivation for the Bayesian approach in estimation. However, the price that is paid is in computational complexity. Specifically, it is necessary to store and to process continuous pdf's. When there is uncertainty in the observation the implementation of the Bayesian approach is further complicated. In using one set of algorithms [11], there is a growing memory requirement. But in certain cases such as Gauss-Markov signals, the pdf's to be stored are characterized by just a few parameters. Therefore, storing and processing them even with growing memory is a manageable task. The algorithms derived in this paper do not require growing memory; so, the pdf's to be stored are relatively few. But in this case, the pdf's, even for Gauss-Markov model, are not Gaussian and hence are not characterized by a few parameters. Therefore, it is necessary to deal with continuous pdf's. This task can, however, be facilitated by discretizing the range spaces of the continuous r.v.'s and using the resulting discretized pdf's as approximations to the continuous ones. The Gauss-Markov example investigated demonstrates that the Bayes smoothing algorithms that are presented yield estimates significantly better than the linear smoother and they are also computationally feasible.

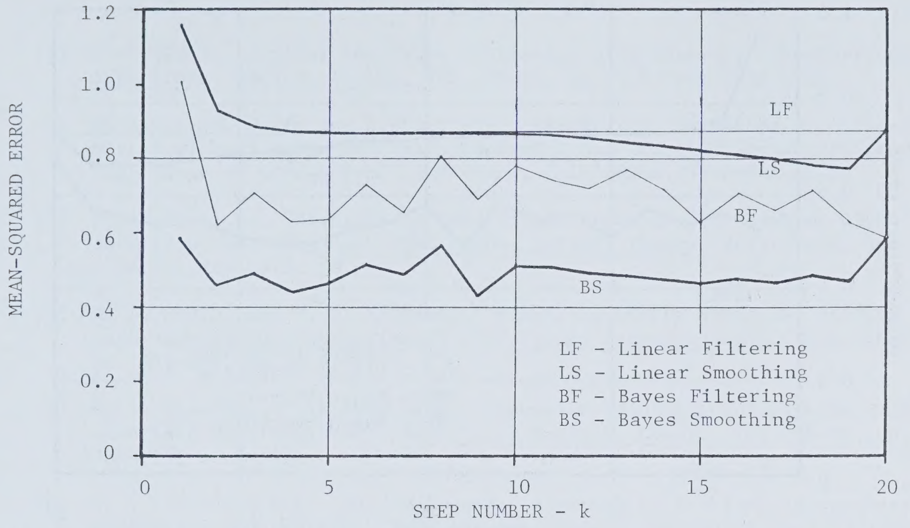


Figure 1. Mean-squared error curves for the linear and Bayes smoothing and filtering estimates ($p_1 = 0.6$, $Q = 1.0$, $R = 0.05$)

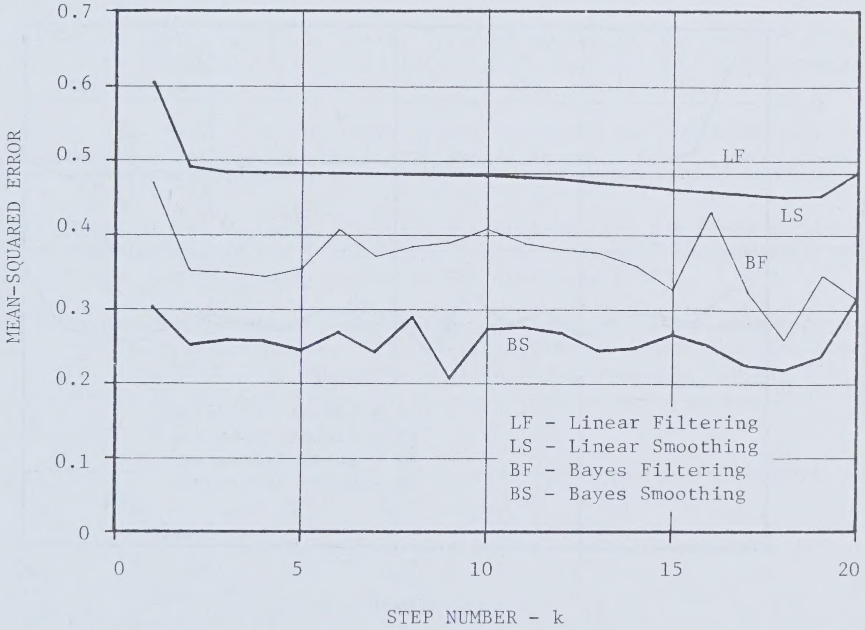


Figure 2. Mean-squared error curves for the linear and Bayes smoothing and filtering estimates ($p_1 = 0.8$, $Q = 1.0$, $R = 0.05$).

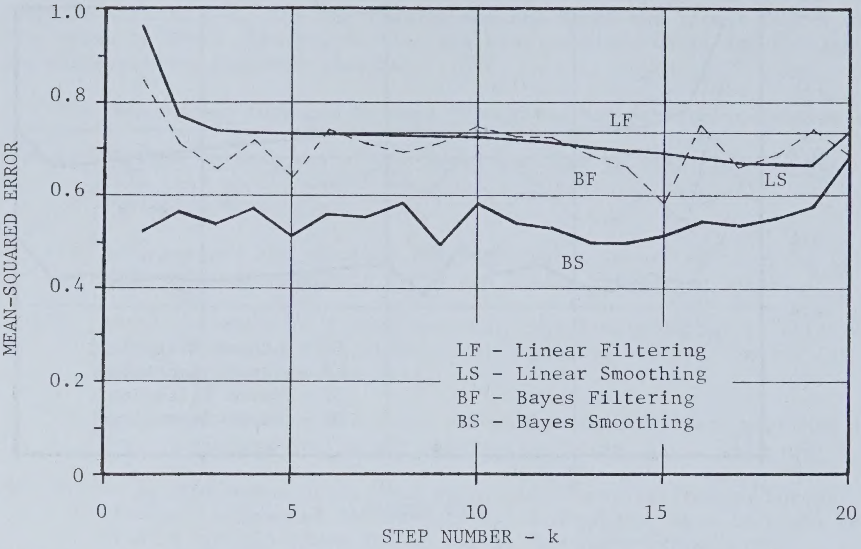


Figure 3. Mean-squared error curves for the linear and Bayes smoothing and filtering estimates ($p_1 = 0.8$, $Q = 1.0$, $R = 0.5$).

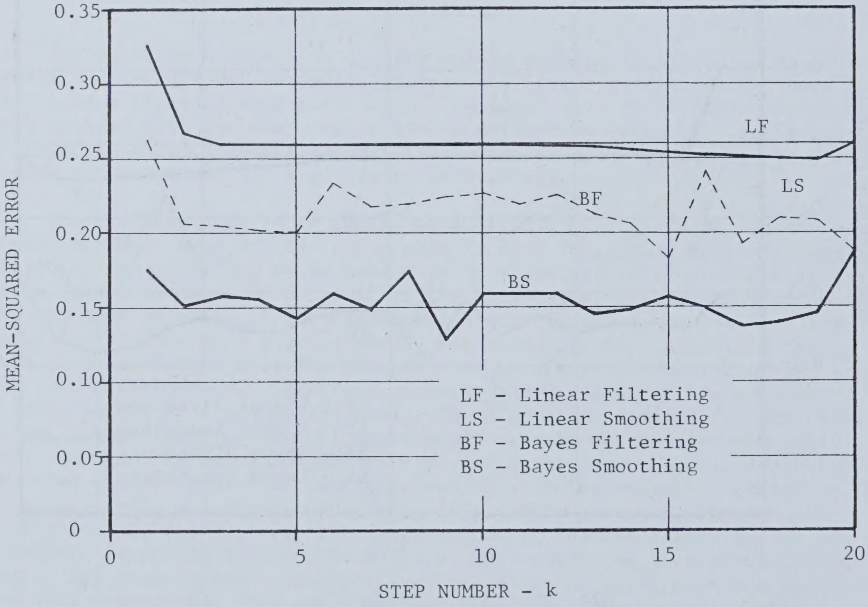


Figure 4. Mean-squared error curves for the linear and Bayes smoothing and filtering estimates ($p_1 = 0.8$, $Q = 0.5$, $R = 0.05$).

REFERENCES

[1] N. E. Nahi, "Optimal recursive estimation with uncertain observations," IEEE Trans. Inform. Theory, Vol. IT-15, pp. 457-462, July 1969.

[2] M. Askar, H. Derin and O. Yurtseven, "Recursive estimation of Gauss-Markov process using uncertain observations," Twelfth Annual Asilomar Conference on Circ., Syst. and Comp., California, November 1978.

[3] R. A. Monzingo, "Discrete optimal linear smoothing for systems with uncertain observations," IEEE Trans. Inform. Theory, Vol. IT-21, pp. 271-275, May 1975.

[4] M. T. Hadidi and S. C. Schwartz, "Linear recursive state estimators under uncertain observations," IEEE Trans. Automat. Contr., Vol. AC-24, pp. 944-948, December 1979.

[5] R. A. Monzingo, "Discrete linear recursive smoothing for systems with uncertain observations," IEEE Trans. Automat. Contr., Vol. AC-26, pp. 754-757, June 1981.

[6] Y. C. Ho and R.C.K. Lee, "A Bayesian approach to problems in stochastic estimation and control," IEEE Automat. Contr., Vol. AC-9, pp. 333-339, October 1964.

[7] A. G. Jaffer and S. C. Gupta, "Recursive Bayesian estimation with uncertain observation," IEEE Trans. Inform. Theory, Vol. IT-17, pp. 614-616, September 1971.

[8] A. G. Jaffer and S. C. Gupta, "Optimal sequential estimation of discrete processes with Markov interrupted observations," IEEE Trans. Automat. Contr., Vol. AC-16, pp. 471-475, October 1971.

[9] M. Askar and H. Derin, "A recursive algorithm for the Bayes solution of the smoothing problem," IEEE Trans. Automat. Contr., Vol. AC-26, pp. 558-561, April 1981.

[10] M. Askar and H. Derin, "Recursive algorithms for the Bayes solution of the fixed-point and fixed-lag smoothing problems," to appear in IEEE Trans. Automat. Contr., Vol. AC-28, September 1983.

[11] M. Askar and H. Derin, "Recursive algorithms for Bayes smoothing with uncertain observations," 1983 American Control Conference, San Francisco, CA, June 1983, also to appear in IEEE Trans. Automat. Contr., Vol. AC-29, No. 6, June 1984.

SIGNAL DETECTION WHEN DEPENDENCE IS NEGLECTED

D.R. HALVERSON
Department of Electrical Engineering
Texas A&M University
College Station, Texas 77843

and

G.L. WISE
Departments of Electrical Engineering and Mathematics
University of Texas at Austin
Austin, Texas 78712

ABSTRACT

We consider the discrete time detection of time varying deterministic signals in weakly dependent noise. We investigate quantitative conditions which allow determining when the dependency can be ignored, and present a result which allows bounding the variation in false alarm rate and detection probability induced by ignoring the dependency. The case of stationary Gaussian noise is then considered, and we show that our result relates to the bound on the variation in false alarm rate and detection probability to the rate of descent of the noise autocorrelation.

I. INTRODUCTION

In this paper we consider the discrete time detection of a deterministic time varying signal in dependent nonstationary noise. If the noise process is independent, then the optimal detection problem is tractably formulated in terms of the Neyman-Pearson criterion [1], which maximizes the detection probability β for a constrained false alarm rate α. However, when dependency is present, application of a Neyman-Pearson approach is often inhibited by lack of knowledge of the required higher order noise distributions. Often in the past, whatever dependency has been present has been ignored in order to obtain tractable results. This has led to variations in the nominal values of α and β due to the residual dependency. If the dependency were weak, then one would hope that these variations would be acceptably small. However, little attention has been directed toward quantitatively determining the extent of the allowable dependency. In this paper we investigate this subject and obtain results which allow bounding the variation in α and β when the detector is designed under an independence assumption with dependency actually present. In particular, our results allow relating the variation in α and β to constraints on the rate of descent of the noise autocorrelation for the Gaussian case.

II. ANALYSIS

Consider a set of (possibly dependent) random variables $Y_1, Y_2, \ldots, Y_n$ whose realizations $y_1, y_2, \ldots, y_n$ represent the observed data. Let P_0 denote the probability measure for the case where the signal is not present, and P_1 the corresponding probability measure for the case where the signal is present (all probability measures will be taken on the Borel sets of $\mathcal{R}^n$). If we were to assume that the $Y_1, Y_2, \ldots, Y_n$ were independent, then the detector would be designed under the auspices of two different measures for the signal not present and signal present cases; we denote these measures by P_0^I and P_1^I respectively.

Suppose now a desired false alarm rate α has been specified. If the detector is designed under an assumption of independent samples, then under the Neyman-Pearson criterion [1] a threshold T is chosen so that when a transformation of the data $\Lambda(y_1, y_2, \ldots, y_n)$ is compared to T, the resultant false alarm rate will be α if the samples are indeed independent. However, as we have remarked, the samples may actually be dependent. Since T is chosen under an independence assumption, the actual false alarm rate α_D (where dependency is present) may vary from the desired rate α. It is clear that it would be helpful to bound the extent of the variation $|\alpha-\alpha_D|$.

In order to do this, we note that the false alarm rate is obtained by evaluating the relevant probability measure at the appropriate Borel set; for example in the common case where a randomized test is unnecessary we have $\alpha = P_0^I\,(\Lambda(Y_1, \ldots, Y_n) \geq T)$ and $\alpha_D = P_0(\Lambda(Y_1, \ldots, Y_n) \geq T)$, thus yielding

$$|\alpha-\alpha_D| = |P_0^I\,(B) - P_0(B)|, \tag{1}$$

where B is the Borel set

$$B = \{(y_1, \ldots, y_n) : \Lambda(y_1, \ldots, y_n) \geq T\}.$$

We are thus motivated to find a bound for (1). For the moment we will confine our attention to the nonrandomized test mentioned above. At the end of this section we will briefly comment on an extension to the randomized case.

It follows from (1) and the proof of Scheffé's Theorem in [2] that

$$|\alpha-\alpha_D| \leq \frac{1}{2} \int_{\mathbb{R}^n} |f_n^0 - f_{n,I}^0| \, d\lambda_0,$$

where f_n^0 and $f_{n,I}^0$ are respective Radon-Nikodym derivatives of P_0 and P_0^I evaluated with respect to a σ-finite dominating measure λ_0 (e.g. $\lambda_0 = (P_0+P_0^I)/2$). Paralleling [3] we then rearrange the above to form

$$|\alpha-\alpha_D| \leq \frac{1}{2} \int_{\mathbb{R}^n-A_0} \left| \frac{f_n^0}{f_{n,I}^0} - 1 \right| f_{n,I}^0 \, d\lambda_0 + \frac{1}{2} \int_{A_0} f_n^0 \, d\lambda_0,$$

where $A_0 = \{(y_1, \ldots, y_n) : f_{n,I}^0 (y_1, \ldots, y_n) = 0\}$.

We then invoke the Schwarz inequality to yield

$$|\alpha-\alpha_D| \leq \frac{1}{2} \left[\int_{\mathbb{R}^n-A_0} \frac{(f_n^0)^2}{f_{n,I}^0} \, d\lambda_0 - 1 \right]^{\frac{1}{2}} + \frac{1}{2} \int_{A_0} f_n^0 \, d\lambda_0.$$

In the case where there exist probability densities in $\mathbb{R}^n$ corresponding to the measures P_0 and P_0^I, the above becomes

$$|\alpha-\alpha_D| \leq \frac{1}{2} \left[\int_{\mathbb{R}^n-A_0} \frac{(f_n^0(x_1, \ldots, x_n))^2}{f_1^0(x_1) \cdots f_n^0(x_n)} \, dx_1 \cdots dx_n - 1 \right]^{\frac{1}{2}}$$

$$+ \frac{1}{2} \int_{A_0} f_n^0(x_1, \ldots, x_n) \, dx_1 \cdots dx_n, \qquad (2)$$

where $f_n^0(\cdot, \cdots, \cdot)$ is the n-dimensional noise density and $f_i^0(\cdot)$ is the univariate noise density of the i-th noise random variable, and where in this case A_0 becomes the set

$$A_0 = \{(y_1, \ldots, y_n) : f_i^0(y_i) = 0 \text{ for some } i = 1, 2, \ldots, n\}.$$

A similar approach can be applied to the detection probability. Let β_I denote the (apparent) detection probability under the independence assumption, and let β_D denote the (actual) detection probability where dependency is allowed. Note that often we can obtain β_I because of tractability of the independence assumption, with the goal of evaluating the detector's performance. However, because of the residual dependency, we really would want to know something about β_D, and a bound on the difference $|\beta_I-\beta_D|$ would thus be desirable. Be methods similar to those before, we obtain

$$|\beta_I-\beta_D| \leq \frac{1}{2}\left[\int_{R^n-A_1} \frac{(f_n^1)^2}{f_{n,I}^1} d\lambda_1 - 1\right]^{\frac{1}{2}} + \frac{1}{2}\int_{A_1} f_n^1 \, d\lambda_1,$$

where f_n^1 and $f_{n,I}^1$ are the respective Radon-Nikodym derivatives of P_1 and P_1^I, evaluated with respect to the σ-finite dominating measure λ_1, and where

$$A_1 = \{(y_1, \ldots, y_n) : f_{n,I}^1 (y_1, \ldots, y_n) = 0\}.$$

In the case where there exist probability densities in R^n corresponding to the measures P_1 and P_1^I we then have

$$|\beta_I-\beta_D| \leq \frac{1}{2}\left[\int_{R^n-A_1} \frac{(f_n^1(x_1, \ldots, x_n))^2}{f_1^1(x_1) \cdots f_n^1(x_n)} dx_1 \cdots dx_n - 1\right]^{\frac{1}{2}}$$

$$+ \frac{1}{2}\int_{A_1} f_n^1(x_1, \ldots, x_n)\, dx_1 \cdots dx_n, \qquad (3)$$

where $f_n^1(\cdot, \cdots, \cdot)$ is the n-dimensional density for $(Y_1, \ldots, Y_n)$ and $f_i^1(\cdot)$ is the univariate density of Y_i, where the $Y_1, \ldots, Y_n$ are governed by the "signal present" assumption, and where $A_1 = \{(y_1, \ldots, y_n) : f_i^1(y_i) = 0 \text{ for some } i = 1, 2, \ldots, n\}$.

Now we turn our attention to the randomized test. Let

$$B_1 = \{(y_1, \cdots, y_n) : \Lambda(y_1, \cdots, y_n) > T\}$$

$$B_2 = \{(y_1, \cdots, y_n) : \Lambda(y_1, \cdots, y_n) = T\}.$$

We recall that if the observation vector lies in B_1 we announce that the signal is present, and if it lies in B_2 we announce with probability p that the signal is present; we note that T and p are chosen via the Neyman-Pearson lemma [1]. In this case (1) becomes

$$|\alpha-\alpha_D| = |P_0^I(B_1) - P_0(B_1) + p(P_0^I(B_2) - P_0(B_2))|.$$

We then use a triangle inequality and mimic the preceding approach to conclude that the upper bounds given in (2) and (3) are multiplied by the factor (1 + p).

III. RESULTS

We will now apply the inequalities (2) and (3) to the case of the detection of a time varying signal $s_1, s_2, \ldots, s_n$ in stationary zero mean Gaussian noise. It then follows (see the Appendix) that the bounds of (2) and (3) are the same, i.e.

$$|\alpha-\alpha_D| \leq \frac{1}{2} \left[\frac{\sigma^{2n}}{[\det(2\sigma^2 R - R^2)]^{1/2}} - 1 \right]^{1/2}$$

$$|\beta_I-\beta_D| \leq \frac{1}{2} \left[\frac{\sigma^{2n}}{[\det(2\sigma^2 R - R^2)]^{1/2}} - 1 \right]^{1/2},$$

whenever $R^{-1} - \frac{1}{2\sigma^2} I$ is positive definite, where σ^2 is the (nonzero) noise variance and R is the (positive definite) noise covariance matrix.

In practice, we would like to relax requirements for exact knowledge of the dependency present. Suppose we consider the class of zero mean stationary Gaussian noises whose autocorrelations R(i) are such that $|R(i)| \leq \sigma^2 e^{-r|i|}$ for some positive real number r. Thus we model "small dependence" with large r. It then follows (see the Appendix) that whenever $r > \ln 3$,

$$|\alpha-\alpha_D| \leq \frac{1}{2} \left| \frac{1}{(1-t^{1/2}) \exp(t^{1/2})} - 1 \right|^{1/2} \tag{4}$$

$$|\beta_I-\beta_D| \leq \frac{1}{2} \left| \frac{1}{(1-t^{1/2}) \exp(t^{1/2})} - 1 \right|^{1/2}, \tag{5}$$

where $t = 2\ [(n-1)\ e^{-2r} + (n-2)\ e^{-4r} + (n-3)\ e^{-6r} + \cdots + e^{-2(n-1)r}]$. We remark that it will be clear from the following examples that the constraint that $r > \ell n\ 3 \cong 1.10$ is really not restrictive since the dependency cannot be ignored for such small values of r.

Let Δ denote the right side of (4) and (5). For various choices of r and n the corresponding value of Δ can be tabulated in the following Table:

Table: Values of Δ for various r and n.

r	n	Δ
2	5	0.163
2	10	0.288
3	5	0.053
3	10	0.081
3	25	0.143
3	50	0.227
4	5	0.019
4	10	0.028
4	25	0.047
4	50	0.069
5	5	0.007
5	10	0.010
5	25	0.017
5	50	0.024
6	5	0.002
6	10	0.004
6	25	0.006
6	50	0.009

Note that as one might expect, Δ is larger for small r or large n, and smaller for large r or small n. Since α is usually chosen reasonably small (typically $\alpha \leq 0.10$) it would thus appear that the residual dependency could not be ignored in most cases for a Gaussian noise process with an autocorrelation whose modulus is lower bounded by $R(i) = \sigma^2\ e^{-r|i|}$ for $r \leq 3$, whereas for r in the range of 4 to 5, the dependency could often be ignored for suitably small sample sizes if $|R(i)| \leq \sigma^2\ e^{-r|i|}$. Moreover, if the dependency falls off sufficiently fast, such as when $|R(i)| \leq \sigma^2 \cdot e^{-r|i|}$ for $r \geq 6$, then we might frequently be able to ignore it for the sample sizes considered here. Note also that even when the dependency cannot be completely ignored, we could still compensate for it if Δ is reasonably small. We would design the detector under an independence assumption and choose a threshold to yield an apparent false alarm rate smaller than $\alpha-\Delta$. We would then be assured that the actual false alarm rate with dependency present would not exceed α.

IV. CONCLUSION

In this paper we have presented a result which allows bounding the variation in false alarm rate and detection probability induced by ignoring the dependency between samples. We have noted that it is often desirable to ignore weak dependence in order to facilitate design of the detector; the bounds we have obtained allow determining when this is possible without overly compromising performance. We then have applied our result to the Gaussian noise case, and have shown that in this situation the bounds reduce to an easily evaluated expression involving the rate of descent of a bound on the noise autocorrelation. This expression conveniently does not require knowledge of the signal for its application.

ACKNOWLEDGEMENT

This research was supported by the Air Force Office of Scientific Research under Grants AFOSR-82-0033 and AFOSR-81-0047.

REFERENCES

1. E.L. Lehmann, Testing Statistical Hypotheses. New York: Wiley, 1970, pp. 65-67.

2. P. Billingsley, Convergence of Probability Measures. New York: Wiley, 1968, pp. 223-224.

3. L. Devroye, "A note on approximations in random variate generation", Journal of Statistical Computation and Simulation, vol. 14, p. 153, February 1982.

4. E. Bodewig, Matrix Calculus. New York: Interscience, 1959.

APPENDIX

Let N_r denote the class of zero mean stationary Gaussian noises whose autocorrelation functions R(i) satisfy $|R(i)| \leq \sigma^2 e^{-r|i|}$. Letting $\bar{x} = (x_1, \cdots, x_n)^T$ and $\bar{s} = (s_1, \cdots, s_n)^T$, and denoting the (nonzero) noise variance and (positive definite) noise covariance matrix by σ^2 and R respectively, we have

$$f_n^0(x_1, \cdots , x_n) = (2\pi)^{-n/2} [\det R]^{-1/2} \exp(- \frac{1}{2} \bar{x}^T R^{-1} \bar{x}) \tag{6}$$

$$f_1^0(x_1) \cdots f_n^0(x_n) = (2\pi)^{-n/2} \sigma^{-n} \exp(- \frac{1}{2\sigma^2} \bar{x}^T I \bar{x}) \tag{7}$$

$$f_n^1(x_1, \ldots, x_n) = (2\pi)^{-n/2} [\det R]^{-1/2} \exp(- \frac{1}{2} (\bar{x}-\bar{s})^T R^{-1} (\bar{x}-\bar{s})) \tag{8}$$

$$f_1^1(x_1) \cdots f_n^1(x_n) = (2\pi)^{-n/2} \sigma^{-n} \exp(- \frac{1}{2\sigma^2} (\bar{x}-\bar{s})^T I(\bar{x}-\bar{s})). \tag{9}$$

Using (6) and (7) we then obtain

$$\frac{(f_n^0(x_1, \ldots, x_n))^2}{f_1^0(x_1) \cdots f_n^0(x_n)} = \frac{\sigma^n}{(2\pi)^{n/2} \det R} \exp (-\bar{x}^T (R^{-1} - \frac{1}{2\sigma^2} I) \bar{x}),$$

whereas (8) and (9) imply

$$\frac{(f_n^1(x_1, \ldots, x_n))^2}{f_1^1(x_1) \cdots f_n^1(x_n)} = \frac{\sigma^n}{(2\pi)^{n/2} \det R} \exp (-(\bar{x}-\bar{s})^T (R^{-1} - \frac{1}{2\sigma^2} I) (\bar{x}-\bar{s})).$$

Applying (2) and (3), we then obtain by direct integration, whenever $R^{-1} - \frac{1}{2\sigma^2} I$ is positive definite,

$$|\alpha-\alpha_D| \leq \frac{1}{2} \left[\frac{\sigma^{2n}}{[\det(2\sigma^2 R-R^2)]^{1/2}} - 1 \right]^{1/2} \tag{10}$$

$$|\beta_I-\beta_D| \leq \frac{1}{2} \left[\frac{\sigma^{2n}}{[\det(2\sigma^2 R-R^2)]^{1/2}} - 1 \right]^{1/2} . \tag{11}$$

Considering now the case where the noise lies in N_r, we note that $\det (2\sigma^2 R-R^2) = \det R \det (2\sigma^2 I-R)$; moreover, R and $2\sigma^2$ I-R are matrices with the same element σ^2 along the diagonal and with the same elements (except for opposite sign) off the diagonal. It therefore follows from [4, p. 81] that

$$|\det R| \geq \sigma^{2n} (1-h^{1/2}) \exp (h^{1/2}), \text{ and} \tag{12}$$

$$|\det (2\sigma^2 I-R)| \geq \sigma^{2n} (1-h^{1/2}) \exp (h^{1/2}), \tag{13}$$

where h is the sum of the squares of the off-diagonal elements of $\frac{1}{\sigma^2} R$. Noting that the right side of (12) and (13) is decreasing for $h > 0$, and noting that the off-diagonal elements of $\frac{1}{\sigma^2} R$ can be bounded via the role of N_r, we have $h \leq t \overset{\Delta}{=} 2[(n-1) e^{-2r} + (n-2) e^{-4r} + \cdots + e^{-2(n-1)r}]$, and thus

$$|\det (2\sigma^2 R-R^2)|^{1/2} = [\det R \det(2\sigma^2 I-R)]^{1/2} \geq \sigma^{2n} (1-t^{1/2}) \exp(t^{1/2}).$$

From (10) and (11) we therefore have, when $R^{-1} - \frac{1}{2\sigma^2} I$ is positive definite,

$$|\alpha-\alpha_D| \leq \frac{1}{2} \left| \frac{1}{(1-t^{1/2}) \exp(t^{1/2})} - 1 \right|^{1/2}$$

$$|\beta_I-\beta_D| \leq \frac{1}{2} \left| \frac{1}{(1-t^{1/2}) \exp(t^{1/2})} - 1 \right|^{1/2} .$$

Finally, we consider the question of the positive definiteness of $R^{-1} - \frac{1}{2\sigma^2} I$. Since R is positive definite, it suffices to investigate $2 I - \frac{1}{\sigma^2} R$. But it follows from the Levy-Hadamard Theorem [4, p. 67] that for noises lying in N_r, $2 I - \frac{1}{\sigma^2} R$ is positive definite if $\frac{2 e^{-r}}{1-e^{-r}} < 1$, i.e. $r > \ell n\ 3$. For the noise class N_r, is thus suffices to replace the positive definiteness condition with simply the requirement that $r > \ell n\ 3$.

DETECTION AND INFORMATION

ANDREW B. MARTINEZ
Department of Electrical Engineering
Tulane University
New Orleans, LA 70118

ABSTRACT

Fisher's Information and Shannon's Mutual Information are considered as they relate to hypothesis testing. Fisher's Information is well known to be related directly to the asymptotic performance of detectors, and it is shown to be a measure of distinguishability rather than of information. Mutual Information is used to give a satisfying description of the inference problem; maximizing the Mutual Information of hypothesis and decision is shown to be equivalent to applying the Neyman-Pearson criterion. While this suggests that Mutual Information can be used to optimize detector sub-systems, this is shown not to be true. The two information measures are compared using an identity relating Entropy and Fisher's Information.

INTRODUCTION

Two concepts of information are considered in this paper as they relate to hypothesis testing: Fisher's Information and Shannon's Mutual Information. Fisher's Information has been used extensively and is well known to be related directly to the asymptotic performance of both detectors and estimators [1-5]. It has recently been suggested that Mutual Information might measure the performance of detector sub-systems, but it is less than clear how well. In this paper it is shown that Mutual Information can give a very satisfying description of the inference problem, but that it does not lend itself to optimization proceedures. While Mutual Information can be said to measure information, it can be said that Fisher's Information measures distinguishability or intelligibility.

The relationship between inference and information was explored by Watanabe [6]. He used Mutual Information to give insight into the process of inference, but he did not consider the optimization of detection schemes using Mutual Information as a measure of detector performance. Woodward [7] considered Mutual Information as it relates to the radar range problem. He found that Mutual Information was insufficient to measure detector performance adequately. Woodward speculated that two quanties must actually be measured, information and intelligibility and that intelligibility is related more to meaning than to information content. This is an interesting point and forms the central theme of this paper. Rather than use the term 'intelligibility', the word 'distinguishability' will be used in this paper.

THE DETECTION PROBLEM

The detection problem consists of using a body of evidence x from a set of possible observations X to decide for one of a set of competing hypotheses, H. A detector is desired to choose one of a set of decisions D based on the observation x.

In general there are one or more auxiliary conditions θ determined by the test situation. In passive observation, the classes of possible observations and auxiliary conditions are determined by the conditions of the test and are beyond the control of the observer. In active experimentation on H, the class of observations and, to a lesser extent, the auxiliary conditions are under limited control. In either case, for a given experiment, both X and θ are fixed.

A binary detection problem is considered in detail in this paper. It can be expressed in terms of an hypothesis H and an alternative H :

H_0: $x \sim f(x,\theta_0)$

H_1: $x \sim f(x,\theta)$ $\qquad \theta \neq \theta_0$

MUTUAL INFORMATION AS A MEASURE OF DISTINGUISHABILITY

In this section, the Mutual Information is used to measure the overall performance of an inference scheme. It is shown that, under certain circumstances, maximizing the information transmission from hypothesis to decision is equivalent to applying the Neyman-Pearson criterion. As a result, the likelihood ratio detector is the maximum Mutual Information detector.

The detector can be viewed as a communication channel with input $H_i \in H$ and output $D_i \in D$. From this point of view, an optimal detector should maximize the information transmitted from H to D. That is, an optimal detector should choose D so that the information about H is maximized.

As a measure of information transfer, consider the Mutual Information:

$$I(D;H) = H(D) - H(D|H)$$

where H(D) is the Entropy of D, and H(D|H) is the conditional Entropy of D given H. Obviously, the Mutual Information is not independent of the observer, as it requires certain a priori knowledge (such as the prior probabilities of H).

For the binary detection problem described above, assume prior probabilties:

$P(H_0) = \zeta$ $\qquad$ and $\qquad$ $P(H_1) = 1-\zeta$

Let D_i be the event "decide for H_i". Then using the traditional notation α for level and β for power of the test:

$P(D_0|H_0) = 1-\alpha \qquad P(D_1|H_0) = \alpha$

$P(D_0|H_1) = 1-\beta \qquad P(D_1|H_1) = \beta$

Thus

$P(D_0) = (1-\alpha)\zeta + (1-\beta)(1-\zeta) = \gamma$

$P(D_1) = \alpha\zeta + \beta(1-\zeta) = 1-\gamma$

The Entropy of D can be found to be

$H(D) = -\gamma \log(\gamma) - (1-\gamma) \log(1-\gamma)$

The joint probabilities of H and D are

$P(D_0,H_0) = (1-\alpha)\zeta \qquad P(D_1,H_0) = \alpha\zeta$

$P(D_0,H_1) = (1-\beta)(1-\zeta) \qquad P(D_1,H_1) = \beta(1-\zeta)$

Thus the conditional Entropy is

$$H(D|H) = -(1-\alpha)\zeta \log(1-\alpha) - \alpha\zeta \log(\alpha) \\ - (1-\beta)(1-\zeta) \log(1-\beta) - \beta(1-\zeta) \log(\beta)$$

The Mutual Information is

$$I(D;H) = -\gamma \log(\gamma) - (1-\gamma) \log(1-\gamma) + (1-\alpha)\zeta \log(1-\alpha) \\ + \alpha\zeta \log(\alpha) + (1-\beta)(1-\zeta) \log(1-\beta) + \beta(1-\zeta) \log(\beta)$$

The inference system is to be optimized by maximizing this quantity.

Assume the a priori probability $P(H_0) = \zeta$ to be beyond the control of the observer. Then the Mutual Information can be shown to be a monotone increasing function of the power as follows. Fix the level α and differentiate the Mutual Information with respect to the power β:

$$\frac{d}{d\beta} I(D;H) = (1-\zeta) \log \frac{\gamma\beta}{(1-\gamma)(1-\beta)}$$

And so

$$\frac{d}{d\beta} I(D;H) > 0 \iff \beta > \alpha$$

Since the derivative is positive, the Mutual Information is a monotone increasing function of β for all $\beta > \alpha$. In this case, maximizing the Mutual Information is equivalent to fixing the level and maximizing the power. Thus the maximum information and Neyman-Pearson criterion are equivalent. It follows that maximizing the information yields the likelihood ratio.

MUTUAL INFORMATION AND DETECTOR SUB-SYSTEMS

The results in the previous section seem to imply that Mutual Information can be used to optimize detector sub-systems. In this section this is shown not to be the case.

Consider the same binary detection problem outlined above. The detector will operate on data $x \in X$ based on hypothesis $H_i \in H$ to decide for $D_j \in D$. A detector sub-system is to be optimized by maximizing the Mutual Information of the sub-system input and output:

$$I(D;X) = H(D) - H(D|X)$$

For a deterministic sub-system, the conditional Entropy is zero and maximizing the Mutual Information is equivalent to maximizing the Entropy of D:

$$H(D) = -\gamma \log(\gamma) - (1-\gamma) \log(1-\gamma)$$

It is easily shown that the maximum occurs at $\gamma = 1/2$. Since $\gamma = P(D_0)$, a maximum Entropy detector is required only to decide for D_0 half the time and D_1 the other half. This is obviously not a particularly good detection scheme. Using the expression for γ given above gives the linear relationship between α and β that the maximum information detector must satisfy:

$$\beta = -\left[\frac{\zeta}{1-\zeta}\right]\alpha + \left[\frac{1}{2(1-\zeta)}\right]$$

Notice that $\alpha = \beta = 1/2$ satisfies the equation and results in an optimal detector which is obviously far from optimal.

FISHER'S INFORMATION AS A MEASURE OF DISTINGUISHABILITY

What is needed is a measure of distinguishability as opposed to a measure of information. Fisher's Information can be shown to be a such a measure. It can be argued that the log-likelihood ratio is a good statistic for discriminating between θ and θ_0 at the point x:

$$l(x) = \log \frac{f(x,\theta)}{f(x,\theta_0)} = \log f(x,\theta) - \log f(x,\theta_0)$$

When θ can be assumed nearly equal to θ_0, then the discrimination rate can be written as:

$$\lim_{\theta \to \theta_0} \frac{\log f(x,\theta) - \log f(x,\theta_0)}{\theta - \theta_0} = \left.\frac{d \log f(x,\theta)}{d\theta}\right|_{\theta=\theta_0} = \frac{f'(x,\theta_0)}{f(x,\theta_0)}$$

Notice that this is the well known score function:

$$s(x,\theta_0) = \frac{f'(x,\theta_0)}{f(x,\theta_0)}$$

The score can be viewed as the sensitivity of the distribution to differentially small changes in the parameter θ at a particular point x.

An average rate of discrimination for all x can be found. If the score is viewed as a random variable, it can be shown to have expected value zero:

$$E\, s(x,\theta_0) = \int_X f'(x,\theta_0)dx = \frac{d}{d\theta} \int_X f(x,\theta_0)dx = 0$$

Thus the variance of the score is used as a measure of overall sensitivity:

$$\mathrm{Var}\ s(x,\theta_0) = \lim_{\theta \to \theta_0} E\,[\frac{d}{d\theta} \log(f(x,\theta))]^2 = I(\theta_0)$$

Notice that the variance of the score is Fisher's Information. Fisher's Information can thus be considered as a measure of the overall sensitivity of a density to diffential changes in a parameter θ.

The efficacy J is often used as a measure of the asymptotic performance of the detector [1-5]:

$$J(T(x)) = \lim_{\theta \to \theta_0} \frac{[\frac{d}{d\theta} E_0(T)]^2}{Var_0 T}$$

The efficacy is maximized resulting in the locally optimal detector with efficacy equal to one over Fisher's Information at $\theta=\theta_0$:

$$J(T) = \frac{1}{I(\theta_0)}$$

The properties of the locally optimal detector are well known, and they have been examined by many authors [1-4].

Fisher's Information is generally considered to be poorly named, since it is not a good measure of information. Many densities can be found with infinite Fisher's Information, and this leads to the conclusion that a single sample has infinite information.

COMPARISON OF INFORMATION MEASURES

A relationship between the two information measures can be derived. Recall that the Entropy of X is given by

$$H(X) = - \int_X f(x,\theta) \log f(x,\theta)\, dx$$

Differentiate with respect to θ to give

$$H'(X) = - \int_X f'(x,\theta) \log f(x,\theta)\, dx - \int_X f'(x,\theta)\, dx$$

The second integral on the right equals zero, thus the second derivative of the Entropy is

$$H''(X) = - \int_X f''(x,\theta) \log f(x,\theta)\, dx - \int_X [f'(x,\theta)]^2/f(x,\theta)\, dx$$

Note that the second integral on the right side of the equation is Fisher's Information. Thus

$$H''(X) = - \int_X f''(x,\theta) \log f(x,\theta)\, dx - I(\theta)$$

This expression can be simplified for two common parameters, location and scale. For a shift in mean:

$$H(x+\mu) = H(x) \quad \Rightarrow \quad H'(x) = H''(x) = 0$$

Thus

$$I(\mu) = - \int_X f''_\mu(x) \log f_\mu(x)\, dx$$

For a change in scale:

$$H_\sigma(x) = H(x) + \log(\sigma) \quad \Rightarrow \quad H''_\sigma(x) = -1/\sigma^2$$

Thus

$$I(\sigma) = 1/\sigma^2 - \int_X f''_\sigma(x) \log f_\sigma(x)\, dx$$

CONCLUSIONS

Inference can be viewed as the transmission of information from hypothesis to decision, with Mutual Information as a logical measure of information transfer. It is shown above that maximizing the Mutual Information is equivalent to applying the Neyman-Pearson criterion to the design of detectors.

When Mutual Information is applied to the design of detector sub-systems, it fails to produce useful systems. The reasons for this are rather straightforward; the detector is not be required to preserve the most information possible in an observation. Only that information directly related to the task of distinguishing between hypotheses is required. To use Woodward's terms, intelligibility rather than information is required.

Fisher's Information, while not measuring information, is shown to be a measure of distinguishability. Such a measure would seem to be useful, and, in fact, it has proved quite useful in the design of both detectors and estimators.

REFERENCES

[1] N.H. Lu and B.A. Eisenstein, "Detection of weak signals in non-Gaussian noise," IEEE Trans. on Information Theory, vol. IT-26, pp. 755-771.

[2] J.H. Miller and J.B. Thomas, "The detection of signals in impulsive noise modeled as a mixture process," IEEE Trans. on Comm., vol. COM-24, May 1976, pp. 559-562.

[3] J.H. Miller and J.B. Thomas, "Detectors for discrete-time signals in non-Gaussian noise," IEEE Trans. on Information Theory, vol. IT-18, March 1972, pp. 241-250.

[4] J.W. Modestingo and A.Y. Ningo, "Detection of weak signals in narrowband non-Gaussian noise," IEEE Trans. on Information Theory, vol. IT-25, pp. 592-600, Sept. 1979.

[5] E.J.G. Pitman, Some Basic Theory for Statistical Inference. New York:Wiley, 1979.

[6] S. Watanaba, Knowing and Guessing, a Quantitative Study of Inference and Information. New York:Wiley, 1969.

[7] P.M. Woodward, "Theory of Radar Information," IRE Trans. on Information Theory, Feb. 1953, pp. 108-113.

ACKNOWLEDGEMENT

This research was supported by the National Science Foundation under grant ECS-8206090.

SELF-ORGANIZING HEURISTICS, IMPLICIT DATA STRUCTURES, AND UNSUCCESSFUL SEARCH* (Extended Abstract)

GREG N. FREDERICKSON
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

ABSTRACT

In [Fs2] the following problem was left open: Consider a set of elements with different weights (access probabilities), and with the intervals between consecutive element values also having weights. Is there a structure using just a constant number of locations in addition to those required for element values, which is self-organizing and supports optimal average unsuccessful search times? The above problem is resolved in the affirmative, and a rather attractive data structure is presented.

1. INTRODUCTION

Additional space, beyond the minimum required to store key values, is often useful in the implementation of various data structures. Consider a data structure to hold a static set of n elements that are to be searched. The elements will be accessed independently according to weights (access probabilities) $p_1, \cdots, p_n$, and the $n+1$ intervening intervals will be accessed according to weights $q_0, \cdots, q_n$. If $2n+1$ pointers are available, then the elements can be stored in an optimal or nearly optimal binary search tree [B, Fm, GW, HT, K1, K2, Me1, Me2]. But without any additional space, how well can one do? In this paper we answer a question left open in [Fs2] concerning unsuccessful search, self-organizing heuristics, and the efficient use of space.

We consider data structures that have only a constant number of additional locations, with the values are stored in the first n locations of a one-dimensional array. Such data structures have been termed *implicit data structures* [MS, Fs1]. Suppose that we know the probabilities of access *a priori*. In an earlier paper [Fs2], we presented implicit data structures and associated search strategies that were considerable improvements over the obvious choice of ordering values by decreasing weight and using sequential search [K2].

Suppose that the elements will be accessed according to fixed probabilities, but these probabilities will not be known in advance. Thus any structure that will have search times sensitive to the probabilities of access must be *self-organizing*. Self-organizing heuristics for sequential search have been studied in

---

* This research was supported in part by the National Science Foundation under Grant MCS-8201083.

[Bi, GMS, H, K2, Mc, R]. The best of these heuristics realize $\Theta(\min\{1/p_j, n\})$ time on average for a successful search. In [Fs2] we used self-organizing heuristics to approximate an implicit structure and showed that two self-organizing heuristics may be used to approximate the structure and realize average successful search times of $O(\log \min\{1/p_j, n\})$.†

In this paper we settle a question left open in [Fs2]. Specifically, we construct an implicit structure that is self-organizing and also handles unsuccessful search well. We show that the average unsuccessful search time for the jth interval is $O(\log \min\{1/q_j, n\})$. Thus our structure and heuristics realize, to within a constant factor, optimal performance.

2. SELF-ORGANIZING HEURISTICS FOR SUCCESSFUL SEARCH

The data structure and heuristics that we present here can be viewed as generalizations of those presented in [Fs2]. We first review implicit structure I from [Fs2], which realizes fast access times for successful search, given that the probabilities of access are known. The elements are partitioned into groups A_i, $i=0,1,\cdots,s$, on the basis of their weights. Let group A_i, $i < s$, be of size $h(i)$, and group A_s be of size at most $h(s)$, where $h(i) = 2^{2^i}-1$. The partition is such that the weight of any element in group A_i is no smaller than the weight of any element in group A_{i+1}, for $i=0,1,\cdots,s-1$. The elements are then arranged in sequential memory by increasing group number, and within group by increasing value.

The search for a desired element in structure I proceeds in the following fashion. Each group is examined in order, starting with group A_0, until the element is found, or the groups are exhausted. Within each group a binary search is performed. Note that individual values of $h(i)$ need not be stored, since they may be computed as needed during the search.

When the probabilities of access are not known, we use implicit structure $\hat{I}$, which is an approximation to structure I. The heuristics will rearrange elements so as to attempt to have the expected position of an element be close to its optimal position. Two heuristics were considered in [Fs2], and we discuss them briefly here, since our new structure will employ generalizations of them.

The first heuristic is an adaptation of the move-to-the-front heuristic [Mc] that is used in structures that are to be sequentially searched. The *move-to-the-first-group* heuristic is as follows: When an element that is not in the first group is referenced, it is moved up to the first group. An element from each intervening group is chosen at random and moved down one group. The second heuristic is an adaptation of the transposition heuristic [Mc] that is also used in structures that are to be sequentially searched. The *move-up-one-group* heuristic is as follows: When an element that is not in the first group is referenced, it is moved up one group, and an element at random in this group is moved down one group.

---

† All logarithms are to the base 2. When $O(\log x)$ is used, we shall mean $O(\max\{1, \log x\})$.

The data rearrangement required by these heuristics can be considerable. In worst case $\Omega(n)$ elements would have to be moved in a self-organizing move. If a self-organizing move were performed after every search, the access times would be degraded significantly, since search times are only $O(\log n)$ in worst case. Instead, a self-organizing move should be initiated only every so often, with the work distributed over succeeding accesses.

Since a self-organizing move cannot in general be completed in one turn, the structure must be left in a state that is not completely well-formed. However, the state should be as close as possible to that already described. This condition may be guaranteed by doing the following. All data movement must be implemented in terms of simple swaps, so that at any point in time all elements are present in the structure. Furthermore, all but two groups should be in completely sorted order, and these two groups should be completely sorted except for one element. Details of the implementation of the heuristics, and analysis of their performance, can be found in [Fs4]. We present some details of the analysis in Appendix A, upon which some results of this paper are based.

3. AN IMPLICIT STRUCTURE FOR UNSUCCESSFUL SEARCH

In this section we present an implicit structure that is self-organizing and handles unsuccessful search well. Elements and intervals in our structure *II* are viewed as paired together. Element j will have a combined probability of access of $p'_j = p_j + q_j$, for $j = 1, \cdots, n-1$, with $p'_n = p_n + q_n + q_0$. Elements are partitioned among groups primarily on the basis of their new weights, with the size of the groups described by the function $\bar{h}(\cdot)$, to be defined later.

For efficient searching, it would be convenient to associate with every value a pointer to the next larger value. Since we can not use pointers, we must guarantee instead that the next larger value is nearby in the structure. Let $\pi(j)$ be the index of the next value larger than v_j in the set. (If v_j is largest, then $\pi(j)$ will index the smallest value.) Precisely stated, we require that if v_j is in group A_r, then $v_{\pi(j)}$ will be in one of the groups A_i, for $i \leq r + 1$. We term this the *patronage property*. If $v_{\pi(j)}$ is in group A_{r+1} and v_j is in group A_r, then v_j is termed the *patron* of $v_{\pi(j)}$, and $v_{\pi(j)}$ is termed the *protégé*. It does not appear to be convenient to be more selective about the location of $v_{\pi(j)}$. For instance, we cannot simply require $v_{\pi(j)}$ to be in group A_{r+1}, since $v_{\pi(j)}$ may have very large weight and thus deserve to be in some group A_i, with $i \leq r$.

We shall describe two search procedures for structure *II*. The first will not be quite as simple, but its correctness will be obvious. The second will be quite simple, but its correctness will be more subtle. The first search is performed as follows. If the search value x is in A_0, then terminate the search with success; otherwise continue to group A_1. To search group A_r, for $r > 0$, do the following. Let w be the largest value smaller than x in A_{r-1}. Search A_r for x. If found, terminate with success. Otherwise, let w' be the smallest value larger than w in $\bigcup_{i=0}^{r} A_i$. If $x < w'$, terminate the search with failure. Otherwise, continue to the next group. The correctness of this procedure is immediate, given that $v_j \in A_{r-1}$ implies that $v_{\pi(j)} \in \bigcup_{i=1}^{r} A_i$.

We simplify the search as follows. Instead of determining w', find w'', where w'' is the largest value smaller than x in A_r. Now terminate the search with failure if $w'' < w$. This termination condition is more convenient, since w'' can be identified as a byproduct in a search for x in A_r.

Lemma 1. The above termination condition is equivalent to $x < w'$. □

If probabilities of access are known, then structure II may be constructed in the following manner. Let the size of group A_i be $h(i)$. Group A_0 will consist of an element with largest weight $p_{\cdot j}$. To construct A_i, given A_{i-1}, do the following. For every v_j in A_{i-1}, insert $v_{\pi(j)}$ in A_i if $v_{\pi(j)}$ is not in $\bigcup_{L=0}^{i-1} A_L$. Complete group A_i with the remaining elements of largest weights $\{p_{\cdot j}\}$. It may be shown that the search time for an element or interval whose weight has rank k is $O(\log k)$.

4. SELF-ORGANIZING HEURISTICS FOR OUR IMPLICIT STRUCTURE

We consider structure $\hat{II}$, which is an approximation to structure II. We consider the move-to-the-first-group and the move-up-one-group heuristics, modified to handle structure $\hat{II}$. The data movement in a self-organizing move will be somewhat more complicated for this structure. The element that is the catalyst for the move will be moved forward in the structure. In addition, it may be necessary to elevate a protégé into the next group down from its patron's group, in order to preserve the patronage property. But this protégé may have to bring along its own protégé, and so forth. (The situation may be roughly analogous to a promotion in an administrative hierarchy, where an appointee is allowed to bring along an associate up to the level directly beneath the appointee's level.) the next lower level.) Which elements will move up in the structure under the move-to-the-first-group heuristic may be determined by the following procedure. Start with j set to the index of the catalyst element, and i set to group index 0. While v_j is not in group A_i, v_j will be moved up, i will be incremented, and j will be set to $\pi(j)$. If i is initially set to the next group up, then this procedure may provide similar information for the move-up-one-group heuristic.

Of course elements must be moved down to make space for the elements that are being moved up in the structure. Two restrictions appear necessary to make the self-organizing structure perform well. First, an element that is a protégé should not be demoted unless its patron is also demoted. If v_L is in A_r and $v_{\pi(L)}$ is in A_{r+1}, and $v_{\pi(L)}$ was demoted but not v_L, then the patronage property would be violated. Second, an element should not be moved down more than one group during all of the self-organizing move.

It is apparent that more than one element must be moved down from certain groups under the move-to-the-first-group heuristic. If the catalyst is not in the first group, then one element will be moved down from the first group. If it has been determined that a elements in group A_{r-1} are to be moved down to group A_r, b elements are to be moved up out of group A_r, and c elements are to be moved up into group A_r, then $a - b + c$ elements should be moved from group A_r down to group A_{r+1}. It is simple to show that no more than $r + 1$ elements from group A_r will be moved down.

The characterization of the self-organizing move given in the preceding paragraphs is sufficient to perform a self-organizing move. From section 2 we know that the self-organizing move cannot be performed all at once, but must be distributed over many subsequent accesses. This means that the self-organizing move must be performed in such a way that it can be interrupted at any point, and only a constant numbers of pointers will be needed to describe the state of the self-organizing move. Furthermore, the partially modified structure must be sufficiently well-formed so that searches may still be performed efficiently, given that the structure is in the middle of a self-organizing move.

The data movement for the move-up-one-group heuristic is simpler, and we consider it first. Let $x = v_j$ be the catalyst, with v_j in group A_L. While x is not in group A_{L-1}, do the following. Find the smallest r such that $v_{\pi^r(j)} \in A_{L+r}$. Thus $y = v_{\pi^{r-1}(j)}$ will be the element to be promoted in A_{L+r-1}. Choose a non-protégé element z at random from A_{L+r-2}. Swap y and z, and restore the order in A_{L+r-1} by a sequence of swaps. When x is in A_{L-1}, restore the order of this group. An example of such a self-organizing move, with the inter-group swaps labeled in order of the execution, is shown in Figure 1. The smallest value of r can be found by a fairly straightforward search. An element v_j can be tested for being a protégé by searching for the position of $v_{\pi^{-1}(j)}$.

We consider searching the not completely well-formed structure while a self-organizing move is under way. Since a protégé is never moved down, at every point during the self-organizing move the patronage property will hold for all elements in the structure. At any point at most two elements (either y and z, or x and z) are out of order in the groups. To search within a group, one must first check to see if the group is out of order. This may be done by seeing if the pointers to x and z are to elements within the group. If for example z is in the group, compare the search value against z. Then binary search the group, being careful not to probe at the position containing z.

We next present a procedure that will guarantee that the structure will be nearly well-formed during a move-to-the-first-group self-organizing move. Let $x = v_j$ be the catalyst. While x is not in group A_0, do the following. Find the smallest r such that $v_{\pi^r(j)} \in \bigcup_{L=0}^{r} A_L$. Let $y = v_{\pi^{r-1}(j)}$. While y is not in group A_{r-1}, do the following. Choose a bounceable non-protégé element z at random from the previous group. Swap y and z, mark z as unbounceable, and restore the order in the group receiving z by a sequence of swaps. When y is in group A_{r-1}, restore the order of this group and mark y as unbounceable. When x is in group A_0, restore the order of this group. Now remove the unbounceable marks from all elements in the structure. An example of such a self-organizing move is shown in Figure 2, again with the inter-group swaps labelled in order of occurrence.

It would be convenient to mark values by having an extra bit per value. Since this may not be possible (and we are interested in implicit structures anyway), the following trick from [Fs3] may be used. Each group may be viewed as partitioned into subgroups of size four. The four elements of a subgroup may be permuted to encode the value of the corresponding four-bit number. The search may handle these permuted subgroups by performing binary search on the first elements of subgroups and then performing a linear search through the two subgroups identified by the binary search. The reader is referred to [Fs3] for a more complete description of this technique.

5. PERFORMANCE OF THE SELF-ORGANIZING HEURISTIC

We now consider the choice of $\tilde{h}(\cdot)$ for structure $I\hat{I}$. In the move-up-one-group heuristic, only elements that are not protégés may be chosen at random to be bounced down one group. Since every element in group A_{i-1} may have a protégé in group A_i, as few as $\tilde{h}(i)-\tilde{h}(i-1)$ elements in group A_i may be bounceable. To guarantee that there are $h(i)$ bounceable elements in group A_i, we choose $\tilde{h}(0) = h(0)$, and $\tilde{h}(i) = \tilde{h}(i-1) + h(i)$ for $i > 0$.

For the move-to-the-first-group heuristic, things are not so simple. In addition to the problem of protégés needing to be protected, in general more than one element may be bounced out of a group. However, we know that at most $i + 1$ elements will be bounced out of group A_i. Hence to guarantee that the probability that a bounceable element j in group A_i is bounced is still no more than $(1-p_j)/h(i)$, we define $\tilde{h}'(0) = h(0)$ and $\tilde{h}'(i) = \tilde{h}'(i-1) + (i+1)h(i)$ for $i > 0$. We now proceed to analyze the performance of these heuristics.

Theorem 1. Let the move-up-one-group heuristic be applied to implicit structure $I\hat{I}$, with $\tilde{h}(\cdot)$ describing the size of the groups. If a self-organizing move is initiated after every $n + cn^{1/2} \log n$ element comparisons in searches, then the expected access times for successful and unsuccessful searches are $O(\log \min\{1/p_j, n\})$ and $O(\log \min\{1/q_j, n\})$, respectively.

Proof. The performance of the heuristic may be modeled by the Markov chain in the proof of Theorem A2, with p_j replaced by p'_j. The probability p'_j that there is a transition from state i to state $i-1$ is an underestimate of the probability that an element is transferred from A_i to A_{i-1}, since an element may be promoted as a result of being a protégé of some element that is promoted. The probability of a transition from state $i-1$ to i remains the same, since it requires that some element other than itself be the catalyst, and that there are $h(i-1)$ elements in group A_{i-1} that can be bounced. Once again this probability is an overestimate, in part because element j may be a protégé and hence not bounceable.

A bound on P_{ji} similar to that in Theorem A2 may thus be established as

$$P_{ji} < \frac{1}{p'_j} \cdot \frac{1}{2^{2^{i-1}}}$$

For the expected search time, given that a self-organizing move is performed after every search, we have

$$C_j \leq \textstyle\sum_{i=0}^{s} P_{ji} \log (\bar{h}(i)+1)$$

From the definition of $\bar{h}(\cdot)$, we have $\bar{h}(i) \leq \frac{4}{3} h(i)$, for $i \geq 0$. Thus the bound on the expected search time, given a self-organizing move after every search, will be $O(\log 1/p'_j)$.

The number of swaps will be no greater than n. The work in testing for protégés is $O(n^{1/2} \log n)$, since there are $O(n^{1/2})$ protégés in groups A_i, for $i < s$. Thus a self-organizing move can be accomplished after the corresponding number of comparisons. Given these results, an analysis similar to that in Theorem A3 establishes that the expected successful search time is $O(\log 1/p'_j)$, which is $O(\log 1/p_j)$. Since the search time for interval j is of order the search time for element j, the expected unsuccessful search time is $O(\log 1/q_j)$. □

We now analyze the move-to-the-first-group heuristic.

Theorem 2. Let the move-to-the-first-group heuristic be applied to implicit structure $\hat{II}$, with $\bar{h}'(\cdot)$ describing the size of the groups. If a self-organizing move is initiated after every $c_1 n \log\log n + c_2 n^{1/2} \log n$ element comparisons in searches, then the expected access times for the successful and unsuccessful searches are $O(\log \min\{1/p_j, n\})$ and $O(\log \min\{1/q_j, n\})$, respectively.

Proof. The proof is similar to that of Theorem 1. The performance of this heuristic may be modeled by the Markov chain in the proof of Theorem A1, with p_j replaced by p'_j. The probability p'_j that there is a transition from state 2 to state 1 is an underestimate of the probability that an element is promoted from $\bigcup_{L=i}^{s} A_L$ to $\bigcup_{L=0}^{i-1} A_L$, since an element may be promoted as a result of being a protégé. The probability of a transition from state 1 to state 2 is an overestimate of the probability that an element is bounced from $\bigcup_{L=0}^{i-1} A_L$ to $\bigcup_{L=i}^{s} A_L$, since at least $i \cdot h(i-1)$ elements in A_{i-1} are bounceable, and at most i elements will be bounced from A_{i-1}.

A bound on P_{ji} similar to that in the proof of Theorem 1 may be derived. For the expected search time, given that a self-organizing move is performed after every search, we have

$$C_j \leq \textstyle\sum_{i=0}^{s} P_{ji} \log (\bar{h}'(i)+1)$$

From the definition of $\bar{h}'(\cdot)$, we have $\bar{h}'(i) \leq (h(i))^2$. Thus the expected search time, given that a self-organizing move is performed after every search, will be

$O(\log 1/p'_j)$.

Since no more than s elements are swapped into any group, and $s < 1 + \log\log n$, the number of swaps is no greater than $n(1 + \log\log n)$. The work in testing for protégés is similar to that in the proof of Theorem 1. The remainder of the proof is similar to that of Theorem 1. □

APPENDIX A.

The analysis in [Fs2] was not complete, and we supply relevant updates here. A self-organizing move will be initiated every n time steps, where a time step will correspond to an element comparison in a search. A self-organizing move may always be completed during this interval.

First, we establish the asymptotic probability that the jth element is in some group A_L with $L \geq i$, given that a self-organizing move is performed after each access. From these probabilities, we determine the expected search time for element j, under the same assumption.

Theorem A1. [Fs2] Let the move-to-the-first-group heuristic be applied to implicit structure $\hat{I}$, with a self-organizing move performed after every search. Then

a. For $i > \left\lceil \log\log 1/p_j \right\rceil$, the asymptotic probability that the jth element is in $\bigcup_{L=i}^{s} A_L$ is less than $\frac{1}{p_j} \cdot \frac{1}{2^{2^{i-1}}}$.

b. The expected search time for the jth element is $O(\log 1/p_j)$. □

Theorem A2. [Fs2] Let the move-up-one-group heuristic be applied to implicit structure $\hat{I}$, with a self-organizing move performed after every search. Then

a. For $i > \left\lceil \log\log 1/p_j \right\rceil$, the asymptotic probability that the jth element is in $\bigcup_{L=i}^{s} A_L$ is less than $\frac{1}{p_j} \cdot \frac{1}{2^{2^{i-1}}}$.

b. The expected search time for the jth element is $O(\log 1/p_j)$. □

We shift to the assumption that a self-organizing move is performed after every nth element comparison. Let $H = \sum_{j=1}^{n} p_j \log 1/p_j$. From [Me1] we have that the expected search time in an optimal binary search tree is no smaller than $H/\log 3$. Using the results of Theorems A1 and A2, it is possible to show that the expected number of searches using a total of n comparisons is larger than $\frac{n}{cH}$. (See [Fs4] for the complete proof.)

Theorem A3. Let a self-organizing move be initiated after every nth element comparison in searches. Then the move-to-the-first-group and the move-up-one-group heuristics, when applied to structure I, support expected access times of $O(\log \min\{1/p_j, n\})$. □

REFERENCES

[B] P. Bayer, Improved bounds on the costs of optimal and balanced binary search trees, Dept. of Computer Science, Mass. Inst. of Technology (1975).

[Bi] J.R. Bitner, Heuristics that dynamically organize data structures, *SIAM J. Comput. 8,* 1 (Feb. 1979) 82-110.

[Fs1] G.N. Frederickson, Implicit data structures for the dictionary problem, *JACM 30,* 1 (Jan. 1983) 80-94.

[Fs2] G.N. Frederickson, Implicit data structures for the weighted dictionary problem, *Proc. 22nd Symp. on Foundations of Computer Science,* Nashville, Tennessee (October 1981) 133-139.

[Fs3] G.N. Frederickson, Implicit data structures for weighted elements, CS-82-04, Department of Computer Science, Pennsylvania State University (1982).

[Fs4] G.N. Frederickson, Self-organizing heuristics for implicit data structures, to appear in *SIAM J. Computing.*

[Fm] M.L. Fredman, Two applications of a probabilistic search technique: sorting $X + Y$ and building balanced search trees, *Proc. 7th Symp. on Theory of Computing* (1975) 240-244.

[GMS] G.H. Gonnet, J.I. Munro and H. Suwanda, Exegesis of self-organizing linear search, *SIAM J. Comput. 10,* 3 (August 1981) 613-637.

[GW] C.C. Gotlieb and W.A. Walker, A top-down algorithm for constructing nearly optimal lexicographic trees, *Graph Theory and Computing,* Academic Press (1972) 303-323.

[H] W.J. Hendricks, An account of self-organizing systems, *SIAM J. Comput. 5,* 4 (Dec. 1976) 715-723.

[HT] T.C. Hu and A.C. Tucker, Optimal computer search trees and variable length alphabetic codes, *SIAM J. Appl. Math. 21* (1971) 514-532.

[K1] D.E. Knuth, Optimum binary search trees, *Acta Informatica 1* (1971) 14-25.

[K2] D.E. Knuth, *The Art of Computer Programming, vol. 3: Sorting and Searching,* Addison-Wesley, Reading, Mass. (1973).

[Mc] J. McCabe, On serial files with relocatable records, *Operations Res. 12* (1965) 609-618.

[Me1] K. Mehlhorn, Nearly optimal binary search trees, *Acta Informatica 5* (1975) 287-295.

[Me2] K. Mehlhorn, Best possible bounds on the weighted path length of optimum binary search trees, *SIAM J. Comput. 6,* 2 (June 1977) 235-239.

[MS] J.I. Munro and H. Suwanda, Implicit data structures for fast search and update, *J. Comp. Sys. Sci. 21* (1980) 236-250.

[R] R.L. Rivest, On self-organizing sequential search heuristics, *Comm. ACM 19,* 2 (Feb. 1976) 63-67.

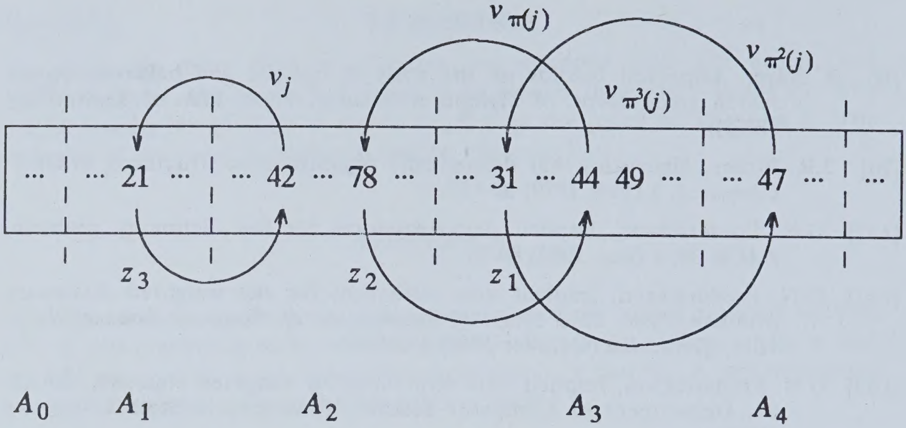

$v_{\pi(j)}$
$v_{\pi^2(j)}$
v_j
$v_{\pi^3(j)}$
... 21 ... 42 ... 78 ... 31 ... 44 49 ... 47 ... ...
z_3
z_2
z_1
A_0
A_1
A_2
A_3
A_4

Figure 1. Inter-group data swaps for structure $I\hat{I}$ under the move-up-one-group heuristic.

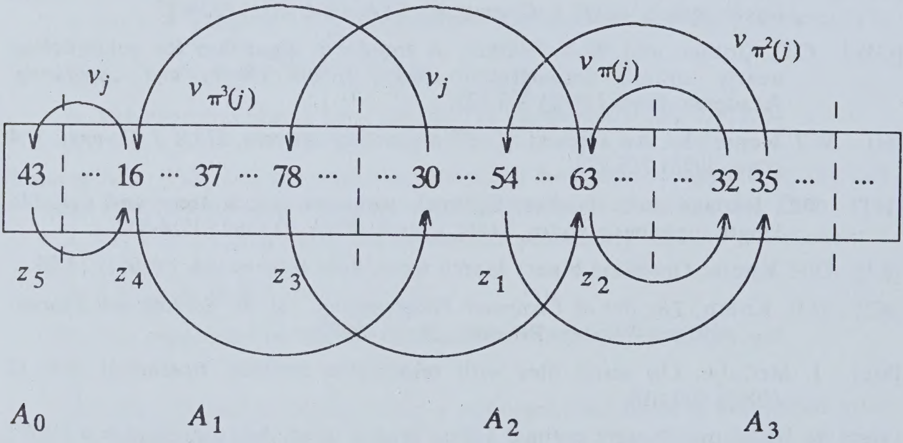

v_j
$v_{\pi^3(j)}$
v_j
$v_{\pi(j)}$
$v_{\pi^2(j)}$
43 ... 16 ... 37 ... 78 ... 30 ... 54 ... 63 ... 32 35 ... ...
z_5
z_4
z_3
z_1
z_2
A_0
A_1
A_2
A_3

Figure 2. Inter-group data swaps for structure $I\hat{I}$ under the move-to-the-first-group heuristic.

ON THE WORST-CASE EFFICIENT IMPLEMENTATION OF WEIGHTED DYNAMIC DICTIONARIES

VIJAY K. VAISHNAVI
Georgia State University
Atlanta, Georgia

ABSTRACT

We show that it is possible to generalize a balanced tree structure like the AVL-trees to the weighted item case, such that the resulting structure retains its restructuring simplicity while providing an efficient implementation of a weighted dynamic dictionary.

INTRODUCTION

The problem of constructing an optimal or a nearly optimal binary search tree for a set of items with known frequencies of access has been addressed by many researchers (e.g., [4], [8]). Knuth [5] posed the more challenging problem of coming up with a "self-organizing" tree structure which "keeps the frequency counts of the searches empirically, maintaining the tree in optimum form depending on the past history of the searches." Bent, Sleator, and Tarjan [1] expanded the scope of the problem by raising the question of the efficient implementation of a "weighted dynamic dictionary." A weighted dynamic dictionary is an abstract data structure for the organization of an ordered list of weighted items which supports the following operations: access, insert, delete, promote, demote, split, and join.

When the weight of an item is actually its frequency of access, it is easy to show, using an entropy argument, that the sum of the weighted access times in a binary search tree is bound below by a constant times $\sum_{i=1}^{n} w_i \log_2 W/w_i$, where w_i is the weight of item i, and $W = \sum_{i=1}^{n} w_i$. Thus the best that can be expected from a binary search tree is "logarithmic behavior," i.e., the access time for item i is $O(\log_2 W/w_i)$. This guarantees minimum average access time of the tree to within a constant factor, i.e., nearly optimal behavior. $O(\log_2 W/w_i)$ has been called [1] the ideal access time for item i.

In the last few years, a number of different tree structures have been suggested for the worst-case efficient (which is distinct from amortized worst-case efficient [10]) implementation of a weighted dynamic dictionary. These structures support ideal access time but are not satisfactory for various reasons. D-trees [9], which are based on weight-balanced trees [5], were originally suggested as a solution to the problem posed by Knuth [5]. These do not seem to support the split operation efficiently. Moreover, these require non-linear storage or else require complicated restructuring operations. The solutions suggested by Gueting and Kriegel [3], Kriegel and Vaishnavi [6], and Bent, Sleator, and Tarjan [2] (2-3 trees with global bias) are based on the 2-3 trees. These solutions do not offer simple direct algorithms for insert, delete, promote, and demote operations. All these solutions can be interpreted as generalizations of certain balanced tree structures (weight-balanced trees, 2-3 trees) such that the items stored have (possibly different) weights associated with them. These generalizations, however, destroy the restructuring simplicity of the base data structures.

In this paper, we show that it is possible to generalize a balanced tree structure like the AVL-trees [5] to the weighted item case, such that the resulting structure retains its restructuring simplicity while providing an efficient implementation of a weighted dynamic dictionary. In the following, we will concentrate on the height increase restructuring algorithm and related results. The other algorithms and results can be found in [12].

WEIGHTED AVL-TREES

A _2-file_ is a collection of (2-dimensional) _vectors_. Each such vector is an ordered pair (x_2,x_1) of _items_; x_i is called an i-item.

T is a _2-dimensional leaf search tree_ for a 2-file F provided: (a) if the root of T, a _2-level node_, is a (2-level) leaf, then it stores x, a 2-item of a vector in F, and points to an _EQSON-subtree_ which is a leaf search tree for the (projection) 1-items obtained by omitting x, the identical 2-items of the vectors; (b) if the root of T is not a leaf, then it stores a suitable key for facilitating the search of the 2-items (in the vectors of F) stored in its 2-level leaves.

The _left neighboring leaf_ of a 2-level node U is the rightmost 2-level leaf to the left of the 2-level leafmost leaf descendant of U. The _right neighboring leaf_ of U is defined similarly.

Let a 2-level node U at height h_1 have V at height h_2 as its left (resp., right) son such that $h_1 > h_2+2$. Then, the two nodes form a _left_ (resp., _right_) _major_ (height) _gap at_ U. The left and the right neighboring leaf nodes of V are the _left_ and the _right neighboring leaf nodes_ of the major gap, respectively. The major gap is _supported_ by one of its neighboring leaf nodes if the height of that node is h_1-2 or more. The major gap is _directly supported_ if it is supported either by its brother or a son of its brother. It is _indirectly supported_ if it is supported but not directly.

A _2-dimensional leaf AVL-tree_ is a 2-dimensional leaf search tree in which every gap is supported, and in which each EQSON-subtree is a leaf height balanced tree with $\Delta = 4$.

Example 1

Figure 1 shows a 2-dimensional leaf AVL-tree for the 2-file $F = \{(1,a),(1,b),(1,c),(2,a),(3,a),(3,b),(4,a),(4,b),(5,a),(5,c),(5,d)\}$. The leaf nodes are shown as square nodes while the non-leaf nodes are shown as circles. Among others, B, C, E are leaf nodes. B and C are 2-level leaf nodes; E is a 1-level leaf. The height of each node is shown beside the node. The left and the right neighboring leaf nodes of C are B and F, respectively. B is, in fact, a brother of C. The nodes A and C constitute the only major gap in the tree.

We now derive an upper bound on the height of a 2-dimensional leaf AVL-tree storing n vectors. An important part of the strategy is to relate the desired bound to that for the height of an "almost 2-dimensional leaf AVL-tree."

An *almost 2-dimensional leaf AVL-tree* (a2AVL-tree) is a 2-dimensional leaf AVL-tree with the following modifications:

(a) The definition of a node U at height h_1 and its son V at height h_2 forming a major gap is relaxed. In the relaxed difinition, they form an *r-gap* only if $h_1 > h_2+5$.

(b) The structure of the tree is constrained by requiring that every r-gap be directly supported.

Lemma 1

Given a 2-dimensional leaf AVL-tree storing n vectors, it can be transformed into an a2AVL-tree of the same height storing n or less vectors.

Proof (Sketch)

A transformation procedure is successively applied at each 2-level node P in decreasing order of the height of the nodes. The procedure transforms T_P, the subtree rooted at P, into T_P of the same height, with the following properties:

(a) There are no major gaps on the leftmost (downward) and the rightmost (downward) paths from the sons of P, which are indirectly supported.

(b) The subtrees rooted at the sons of P are 2-dimensional leaf AVL-trees.

(c) T_P is an a2AVL-tree except that the vectors may not be stored in the right order.

(d) The number of vectors stored in $\overline{T}_P$ is less than or equal to the numbers of vectors stored in T_P. We omit the description of the procedure [12].

Let N(h,2) and N(h,1) be the minimum number of vectors stored in an a2AVL-tree and a leaf height balanced tree with $\Delta = 4$, respectively.

Lemma 2

$N(h,2) \geqslant N(h-3,1)+1, \ h \geqslant 2.$

Proof (By induction on h)

Theorem 1

An upper bound on the height of a 2-dimensional leaf AVL-tree storing n vectors is $\lfloor 2.464\log_2(n+0.5)+2.44 \rfloor$.

Proof

Follows from Lemmas 1, 2, and [7].

We are now in a position to define a weighted AVL-tree:

A *weighted AVL-tree* for a set S of n items with item i having weight w_i is a 2-dimensional leaf AVL-tree whose 2-level leaves store the items in S such that the EQSON-subtree of each 2-level leaf L_i storing item i points to an EQSON-subtree with $\lceil w_i \rceil$ number of (1-level leaves), with the modification that the L_i is placed at height $\lfloor 2.464\log_2(\lceil w_i \rceil +1.5)-0.56 \rfloor +1$, and its EQSON-subtree is deleted.

Theorem 2

A weighted AVL-tree has ideal access time for all the items.

Proof (Sketch)

The maximal height of a 2-dimensional leaf AVL-tree storing n items with weight w_i, $1 \leqslant i \leqslant n$, is $\lfloor 2.464 \log_2(\overline{W}+0.5)+2.44 \rfloor$, where $\overline{W} = \sum_{i=1}^{n} \lceil w_i \rceil$ (see Theorem 1). An upper bound on the height of the EQSON-subtree pointed to by L_i storing item i is $\lfloor 2.464 \log_2(\lceil w_i \rceil +1.5)-0.56 \rfloor$ [7]. The result follows.

HEIGHT INCREASE RESTRUCTURING AND RELATED RESULTS

The following procedure transforms a weighted AVL-tree T, which may have become invalid because the height of its leaf node P has increased by x, $x \geqslant 1$, into a valid weighted AVL-tree.

```
procedure height_increase (P:leaf; x:positive_integer);
   procedure height_increase_restructuring (B:node; y:positive_integer);
   {If B is not the root of T, then the procedure first modifies the
   subtree rooted at A, the father of B, which may or may not involve
   rebalancing, and then possibly calls the procedure recursively at A}
   begin {height_increase_restructuring}
      let the height of A (the father of B), if A exists, be increased
      by z, $0 \leqslant z \leqslant y$;
      if B is not the root of T then
         if (a new major gap is created at A but is directly supported)
            or (a new major gap is not created at A) then
            begin
               if $z \geqslant 1$ then height_increase_restructuring(A,z)
            end
         else
            begin
               apply an appropriate rebalancing operation shown in
               Figures 2 and 3, and their symmetrical variants, at A
               {see Notes (a) and (b)}; let Q be the root of the
               modified subtree;
               if (z = 1) then
                  begin
                     if the rebalancing does not cause height decrease
                     then
                        height_increase_restructuring(Q,1)
                  end
               else height_increase_restructuring(Q,z-1)
            end
   end {height_increase_restructuring};
begin {height_increase}
   height_increase_restructuring(P,x)
end {height_increase};
```

Notes

(a) The rebalancing operation shown in Figure 2, or its symmetric variant, is the rotation. In Figure 2(a), the height of B has increased to $h-1+z$ ($z \geqslant 1$) which in turn has increased the height of A from h to $h+z$.

(b) The rebalancing operation shown in Figure 3, or its symmetric variant, is the double rotation.

Lemma 3

The following is an invariant property of the height increase restructuring procedure: the modified version of the subtree rooted at A is a valid weighted AVL-tree. If $z = 1$, then its height stays the same or decreases by 1. If $z > 1$, then its height decreases by 1 and it looks like one of the subtrees shown in Figure 4 or their symmetric variants.

Proof (Sketch)

If the modified version of the subtree rooted at A is not the result of rebalancing, then the result is obviously true. Otherwise, the procedure must be entered because of a recursive call.

Case 1 Rotation is applied.

1.1 $z = 1$. The only case when a new major gap (between B and A) is created in the rebalanced tree is when $p \leqslant h-3$, $r \leqslant h-3$. But then there is a left major gap at A (Figure 2(a)) even without height increase, and the leaf node that supports this major gap will also support the left major gap at A (Figure 2(b)).

1.2 $z > 1$. By the invariant property, either the right son of B (Figure 2(a)) is a leaf at height $h-2+z$ or the left son of the right son of B is at height $h-3+z$. In either case, the left major gap at B (Figure 2(b)) is directly supported. Hence the result.

Case 2 Double Rotation is applied.

The proof is similar to that of Case 1.

The following results easily follow:

Lemma 4

The height increase procedure is correct.

Lemma 5

Time complexity of the height increase procedure is O(original height of T + x).

Theorem 3

The time needed for updating a weighted AVL-tree after a search is performed is proportional to search time.

Let item i have weight w_i. Let W and $\overline{W}$ be the sum of the weights of all the items stored before and after an operation. Let the weighted dynamic dictionary be implemented as a weighted AVL-tree.

Theorem 4

(a) Search of item i can be performed in time $O(\log_2 W/w_i)$.

(b) Insertion of an item can be performed in time $O(\log_2 W)$.

(c) Promotion of item i to $w_i+\delta$ can be performed in time $O(\log_2 \overline{W}/w_i)$.

CONCLUDING REMARKS

The following additional results are essentially based on a "height decrease" procedure which is similar to the height increase procedure discussed above:

(a) Deletion of the item i can be performed in $O(\log_2 W/w_i)$ time.

(b) Demotion of item i to $w_i-\delta$ can be performed in time $O(\log_2 W/(w_i-\delta))$.

(c) Split of the dictionary at item i can be performed in time $O(\log_2 W/w_i)$.

(d) If h_1 and h_2 are the heights of the weighted AVL-trees for the dictionaries D_1 and D_2, respectively, then the join of the dictionaries can be performed in time $O(h_1-h_2)$.

We have provided a simple solution for the worst-case-efficient implementation of a weighted dynamic dictionary. This is based on a "natural" 2-dimensional generalization of the leaf AVL-trees. The nature of such generalization [11], in fact, provides a methodology for obtaining similar solutions based on other balanced tree structures as well.

REFERENCES

1. S. W. Bent, D. D. Sleator, and R. E. Tarjan, "Biased 2-3 trees," in Proc. 21st Annual IEEE Symposium on Foundations of Computer Science (1980), 248-254.

2. S. W. Bent, D. D. Sleator, and R. E. Tarjan, "Biased search trees," Technical Memorandum, Bell Laboratories, 1983.

3. H. Gueting and H. P. Kriegel, "Dynamic k-dimensional multiway search under time-varying access frequencies," in Proc. 5th GI Conf. Theoretical Computer Science, Lecture Notes in Computer Science, Vol. 104, pp. 135-145, 1981.

4. Knuth, D. E., "Optimum binary search trees," Acta Informatica, Vol. 1, pp. 14-25, 1971.

5. D. E. Knuth, The Art of Computer Programming, Vol. III: Sorting and Searching, Addison-Wesley, Reading, Mass., 1973.

6. H. P. Kriegel and V. K. Vaishnavi, "Weighted multidimensional B-trees used as nearly optimal dictionaries," in Proc. 10th International Symposium on Mathematical Foundations of Computer Science, Vol. 118, pp. 410-417, 1981.

7. F. Luccio and L. Pagli, "On the height of height balanced trees," IEEE Trans. Computers, Vol. C-25, pp. 87-90, 1976.

8. K. Mehlhorn, "Nearly optimal binary search trees," Acta Informatica, Vol. 5, pp. 287-295, 1975.

9. K. Mehlhorn, "Dynamic binary search," SIAM Journal of Computing, Vol. 8, pp. 175-198, 1979.

10. D. D. Sleator, and R. E. Tarjan, "Self-adjusting binary search," Proc. 15th Annual Symposium on Theory of Computing, pp. 235-245, 1983.

11. V. K. Vaishnavi, "Multidimensional height-balanced trees," Technical Report No. 19, Department of Information Systems, Georgia State University, 1983.

12. V. K. Vaishnavi, "Weighted AVL-trees," Technical Report No. 22, Department of Information Systems, Georgia State University, 1983.

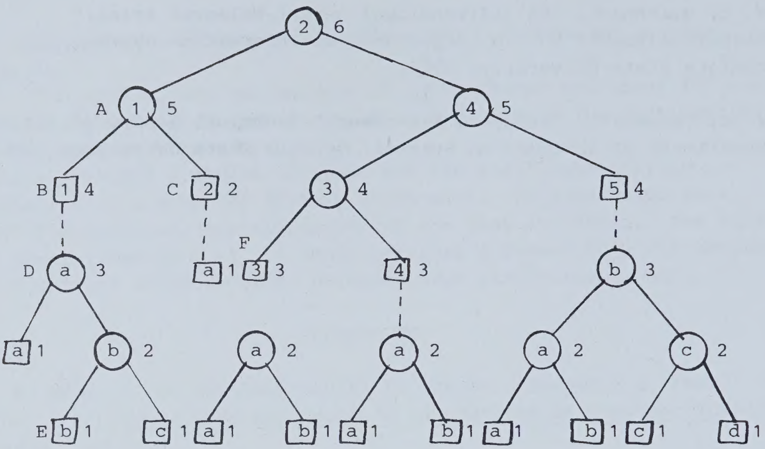
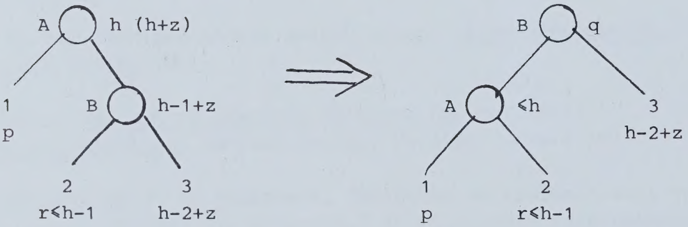

2 6
A 1 5
4 5
B 1 4
C 2 2
3 4
5 4
F
D a 3
a 1
3 3
4 3
b 3
a 1
b 2
a 2
a 2
a 2
c 2
E b 1
c 1
a 1
b 1
a 1
b 1
a 1
b 1
c 1
d 1

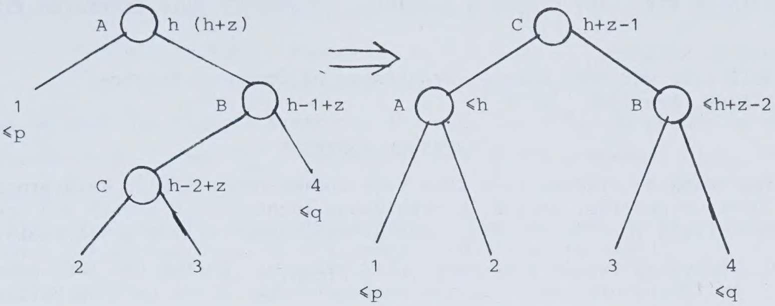
Figure 1


A h (h+z)
1
p
B h-1+z
2
r≤h-1
3
h-2+z
B q
A ≤h
3
h-2+z
1
p
2
r≤h-1

(a) (b)

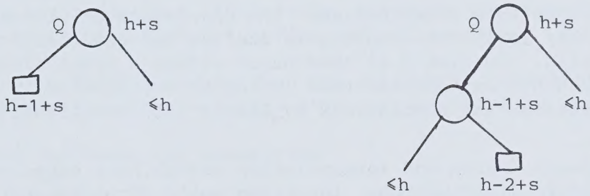
```
if z = 1 then p = h-2; q = h or h+1 end
else begin p = h-1; q = h+z-1 end
```

Figure 2


A
h (h+z)
1
≤p
B
h-1+z
C
h-2+z
4
≤q
2
3
C
h+z-1
A
≤h
B
≤h+z-2
1
≤p
2
3
4
≤q

```
if z = 1 then begin p = h-2; q = h-2 end
else begin p = h-1; q = h-1 end
```

Figure 3


Q
h+s
h-1+s
≤h
Q
h+s
h-1+s
≤h
≤h
h-2+s

s = z-1 or z

Figure 4

SURPRISINGLY EFFICIENT SEARCH ALGORITHMS FOR NONUNIFORMLY GENERATED FILES

DAN E. WILLARD
Bell Labs and SUNY Albany, Department of Computer Science,
Albany, New York 12222

EXTENDED ABSTRACT

Searching an ordered file is a very common operation in data processing. Given N records, stored in contiguous locations in memory and ordered by numeric keys $Y_1 < Y_2 \ldots < Y_N$, one often wishes to find a particular record whose key equals some particular value α. During the last seven years, the literature has developed two approaches for attaining better CPU times than binary search. The first assumes that each key is a positive integer less than some specified constant M, and it studies how to guarantee worst-case time with an index. Various different complexities [4,9,11,12] have been derived for this problem depending on the memory space of the index and the type of query performed.

A second literature [3,6,7,14] has studied searching when no index is present and when the keys are arbitrary real numbers. In this case, the goal is to optimize expected cost using the fact that the keys are stored in sequential order. These articles have shown that an algorithm called interpolation search will guarantee the very tight expected time bound $O(1) + \log \log N$ when the keys are generated by the uniform distribution. These results do not generalize to nonuniform distributions unless one uses the distribution function $D(y) = \int_{-\infty}^{y} u(x)\, dx$ to map an initial nonuniform distribution onto a uniform distribution. The disadvantages of such mappings is that they implicitly constitute indices and can therefore be expensive to maintain. Section 2 of this paper offers a brief proof that interpolation search has an expected time $\Omega(\log N)$ on essentially all nonuniform distribution, and simulations by Gonnet [3] have illustrated a similar point.

Since both binary search and interpolation search have expected times $\Omega(\log N)$ on nonuniform distributions, intuition would first suggest that the same would be true for a hybrid algorithm that simply alternates between these two searching methods. To the contrary, this extended abstract proves such a hybrid has an expected time $\sqrt{\log N}$ on essentially all nonuniform distributions.

The methods developed in this paper also lead to more complicated algorithms with expected time $O(\log \log N)$ on the files generated by nonuniform densities [10]. These algorithms must make probes lying properly between the cut indices of interpolation search and binary search. We have omitted a description of such algorithms in this abstract because the main point can be understood without them.

1. STATEMENT OF RESEARCH PROBLEM

An ordered file of cardinality N will be said to be generated by the probability density u iff this file is constructed by taking N records whose keys are independently determined by u and storing these records in ascending order. The probability density u will be said to be regular iff there exists some vector (b_1,b_2,b_3,b_4) such that u and its first derivatives satisfy the following conditions:

$$u(y) \geq b_1 > 0 \quad \text{whenever} \quad b_3 < y < b_4 , \tag{1}$$

$$|u'(y)| \leq b_2 \quad \text{whenever} \quad b_3 < y < b_4 , \tag{2}$$

$$u(y) = 0 \quad \text{whenever} \quad y \leq b_3 \text{ or } y \geq b_4 , \tag{3}$$

We assume that the file of N records ($Y_1 < Y_2 \ldots < Y_N$) is padded by the two psuedo-keys, Y_0 and Y_{N+1}, containing u's bounding vectors of b_3 and b_4, to simplify the notation.

All algorithms for searching ordered files can then be regarded as iterative procedures whose i-th iteration searches the subfile $F_i = (Y_{L_i} < \ldots < Y_{R_i})$ by generating a cut index C_i and comparing the cut value Y_{C_i} with the search key α. If $Y_{C_i} = \alpha$ then the search terminates successfully; otherwise, the next iteration will examine either the $(Y_{L_i} < \ldots < Y_{C_i})$ or $(Y_{C_i} < \ldots < Y_{R_i})$ subfile according to whether or not $Y_{C_i} < \alpha$.

Under binary search [4] the cut index is the middle position in subfile $F_i = (Y_{L_i} < \ldots < Y_{R_i})$. It can therefore be defined as:

$$C_i^{BIN} = \left\lceil (L_i + R_i)/2 \right\rceil . \tag{4}$$

Under interpolation search [6] a comparable request for the key α will produce a cut index that intuitively represents the expected position of α assuming that the untested interior keys of subfile $F_i = (Y_{L_i} < \ldots < Y_{R_i})$ are uniformly distributed between the previously tested boundary keys Y_{L_i} and Y_{R_i}. In order to define this concept formally, let N_i and ℓ_i denote the following two quantities:

$$N_i = R_i - L_i - 1 , \tag{5}$$

$$\ell_i = Y_{R_i} - Y_{L_i} . \tag{6}$$

In this notation, the cut index, C_i^{INT}, of interpolation search is

$$C_i^{INT} = \left\lceil L_i + \frac{\alpha - Y_{L_i}}{\ell_i} N_i \right\rceil . \tag{7}$$

Our new algorithm, called ALTERNATE, will simply be a hybrid, which sets the C_i equal to C_i^{INT} when i is an odd numbered iteration, and to C_i^{BIN} when i is even. The main theorem of this paper will state that ALTERNATE has an expected time $O(\sqrt{\log N})$ on all regular probability density with only its coefficient depending on the vector (b_1,b_2,b_3,b_4) bounding u.

2. THE NEED FOR ALTERNATE

The analog of interpolation search in numerical analysis is called the method of regula falsi. Given a function g and a value y, this method consists of an iterative procedure that conducts repeated interpolations to find an approximate solution for $g(x) = y$. The method of resula falsi is known to require $\Omega[\log (1/\varepsilon)]$ runtime to make approximations with an accuracy of ε [8].

It is easy to show this result implies that interpolation search has a logarithmic lower bound on its expected runtime for searching nonuniformly generated files, since interpolation search has to overcome all the obstacles encountered by regula falsi plus the additional problem of noise associated with randomly generated files. Precise analogs of the algorithm ALTERNATE do not appear in the numerical literature because those procedures would also require $\Omega(|\log N|)$ time for finding a bracketting interval of length $1/N$ around the root. However, the nearer of the two bracketting pointers should fall within a distance $1/N$ of the root after $\sqrt{\log N}$ iterations. Since interpolation search uses time $\log \log N$ for searching uniform probability distributions, it is natural to conjecture that the sum of the last two quantities, i.e., $O(\sqrt{\log N})$, would be ALTERNATE's expected search time on any nonuniform regular density. The next section will introduce a formalism for proving this conjecture.

3. FORMAL ANALYSIS

Recall that the symbol $F_i = (Y_{L_i} < \ldots < Y_{R_i})$ designates the portion of the file F which will be searched by the i-th iteration of ALTERNATE. The vector $(L_i, R_i, Y_{L_i}, Y_{R_i})$, will be called the state of this iteration. Two attributes of this state are the quantities N_i and ℓ_i (from page 2), three others are defined below:

$$\varepsilon_i^L = \alpha - Y_{L_i} \quad , \tag{8}$$

$$\varepsilon_i^R = Y_{R_i} - \alpha \quad , \tag{9}$$

$$\varepsilon_i = \{\mathrm{MIN}(\varepsilon_i^L, \varepsilon_i^R) \text{ if } N_i \geq 1 \text{ and } 0, \text{ otherwise}\} \ . \tag{10}$$

One final prefatory remark: Let D_i denote the ratio $\varepsilon_i N_i / \ell_i$. Using slightly different notation*, Yao and Yao [12] have noted that $D_{i+1} < D_i$ whenever the i-th cut index is the interpolation mark C_i^{INT}, and they have used this fact to greatly simplify their proofs. Usually this inequality will also hold when the cut index is C_i^{BIN}, but the latter cut index does not guarantee D's decrease. This fact complicates our analysis because the proof would be significantly easier if one could identify a single attribute which always decreases. Our proofs are based around the attribute ε^L, ε^R, ℓ, and N instead of D because a cut index C^{BIN} and C^{INT} always decreases them. The interaction between these four attributes is complicated, but it is easier than basing a proof around an attribute D which does not always decrease.

Throughout this paper, we let N and ℓ denote the initial values assumed by these vectors, that is, they are abbreviations for N_1 and $\ell_1 = b_4 - b_3$. Also, if i is an odd number then ALTERNATE's i-th 2-cycle will be defined as the change in state between the iterations i and i+2.

---

*They denote D as αN; we use the symbol D because we prefer [6]'s notation.

LEMMA 1. A sufficient condition for ALTERNATE to have an expected time $O[f(N)]$ is that there exist a class of events Π and a constant P such that

i) ε_i is always bounded by $O(\ell/N^2)$ after the occurrence of $f(N)$ events belonging to the class Π; and

ii) the i-th cycle will belong to Π with a probability $> P$.

Proof Sketch. The distance between α and its nearest neighbor in the file F will be $< \ell/N^2$ with a probability $< O(1/N)$, by a simple application of the theory of order statistics [5] to regular probability densities. Since $O(1/N)$ bounds the probability of some iteration satisfying $0 < \varepsilon_i < \ell/N^2$, and since ALTERNATE terminates precisely when $\varepsilon_i = 0$ and it never requires more than time $O(N)$ to probe a file, its expected time for a full search exceeds the time to reach a state where $\varepsilon_i < \ell/N^2$ by an inconsequential amount $\leq O(1)$. Since assumptions (i) and (ii) of our lemma imply the latter time is $\leq f(N)/P$, the previous sentence implies that $O(f(N))$ also bounds the time for a full search. Q.E.D.

Let r and s be constants satisfying $0 < r < 1 < s$. Define a 2-cycle to be strongly reducing with respect to (r,s) iff it satisfies one of the following conditions:

i) $N_i \leq 2^{s\sqrt{\log N}}$;

ii) $\ell_{i+2} \leq r\ell_i$ and $\ell_i \geq \ell/2^{s\sqrt{\log N}}$;

iii) $\varepsilon^L_{i+2} \leq \varepsilon^L_i \cdot 2^{-\sqrt{\log N}}$

iv) $\varepsilon^R_{i+2} \leq \varepsilon^R_i \cdot 2^{-\sqrt{\log N}}$

We will prove ALTERNATE has an expected time $\sqrt{\log N}$ by applying Lemma 1 to the class of events defined above. The following lemma shows $f(N)$ lies in $O(\sqrt{\log N})$ for this class:

LEMMA 2. For each r and s there exists a constant K such that a sequence of $K \cdot \lceil\sqrt{\log N}\rceil$ strongly reducing cycles of ALTERNATE will necessarily cause $\varepsilon_i < \ell/N^2$.

Proof Sketch. Each of the vectors N_i, ℓ_i, ε^R_i and ε^L_i decreases monotonically as i increases. This implies no more than $s\lceil\sqrt{\log N}\rceil$ events of type-(i) may occur, since the first of these events happens after $N_i < 2^{s\sqrt{\log N}}$ and the binary cut in each 2-cycle of ALTERNATE reduces this quantity by a factor of 2. Similarly, it is impossible for more than $\lceil s\lceil\log_r 2\rceil \lceil\sqrt{\log N}\rceil\rceil$ cycles of type-(ii) to occur before such a sequence will cause the vector ℓ_i to drop from its initial value of ℓ to a final value underneath the lower bound $\ell/2^{s\sqrt{\log N}}$. Therefore if $K = 2 + s + \lceil s\lceil\log_r 2\rceil\rceil$, then a sequence of $2K\lceil\sqrt{\log N}\rceil$ strongly reducing cycles will include at least $2\lceil\sqrt{\log N}\rceil$ cycles of either the types (iii) or (iv). The latter is a sufficiently large quantity to force either ε^L_i or ε^R_i to fall beneath ℓ/N^2. Q.E.D.

THEOREM 1. The algorithm ALTERNATE has an expected time $O(\sqrt{\log N})$ for any regular probability density (with a coefficient naturally depending on the vector (b_1,b_2,b_3,b_4) bounding u).

Proof Sketch. Let i be an odd number, C_i^{INT} the interpolation index produced by this iteration, and E_1, E_2, and E_3 the following terms:

$$E_1 = \sqrt{\varepsilon_i^L \varepsilon_i^R / N_i} \; , \tag{11}$$

$$E_2 = \varepsilon_i^L \varepsilon_i^R \; , \tag{12}$$

$$E_3 = |(\varepsilon_i^R - \varepsilon_i^L)/N_i| . \tag{13}$$

Then we claim there will exist a constant a whose value depends only on u's bounding vector such that

$$\mathrm{Prob}\{|Y_{C_i^{INT}} - \alpha| < a(E_1+E_2+E_3)\} > \frac{4}{5} \tag{14}$$

Equation (14) follows essentially because E_1, E_2, and E_3 are asymptotic estimates of errors due to the randomness of the ordered file, the non-uniform nature of u (i.e., equations (1) through (3)), and the integer rounding in the calculation of C_i^{INT} (equation 7), respectively. It is also known [5] that order statistics converge to normal distributions when their ballot numbers are chosen from the uniform density. For a non-uniform density u, the translation of this result implies the existence of two further constants, d_1 and d_2, which guarantee equation (15) if $\varepsilon_i^R \geq \varepsilon_i^L$, and its mirror image otherwise. (See Appendix on next page.)

$$\mathrm{Prob}\{Y_{C_i^{INT}} > \alpha + d_1(E_1+E_3) - d_2E_2\} > \frac{2}{5} \tag{15}$$

Subtracting equation (15) from (14) we obtain equation (16) and its mirror image.

$$\mathrm{Prob}\{\alpha + d_1(E_1+E_3) - d_2E_2 < Y_{C_i^{INT}} < \alpha + a(E_1+E_2+E_3)\} > \frac{1}{5}. \tag{16}$$

Equations (11) through (13) imply that there exists a constant s whose value depends on a, d_1, and d_2 such that if $\varepsilon_i^R < 2^{-s\sqrt{\log N}}$, $\varepsilon_i^L < 2^{-s\sqrt{\log N}}$, and $N_i > 2^{s\sqrt{\log N}}$ then the following inequalities will hold.

$$a(E_1+E_2+E_3) < 2^{-\sqrt{\log N}} \mathrm{MAX}(\varepsilon_i^R,\varepsilon_i^L) \tag{17}$$

$$a(d_2+d_1)E_2/d_1 < 2^{-\sqrt{\log N}} \mathrm{MIN}(\varepsilon_i^L,\varepsilon_i^R) \tag{18}$$

Now we will use these facts to prove that every iteration of ALTERNATE has at least a probability 1/5 of being strongly reducing with respect to the constant s defined above and a constant r to be introduced soon. This fact, whose proof is divided into four cases, immediately implies ALTERNATE has an expected time $O(\sqrt{\log N})$, by Lemmas 1 and 2.

Verification for the Case where $\ell_i < 2^{-s\sqrt{\log N}}$, $N_i > 2^{s\sqrt{\log N}}$, and $E_2 \geq (d_2/d_1)(E_1+E_3)$. The last inequality combined with equation (14) implies there is a probability $\geq 4/5$ that $|Y_{C_i^{INT}} - \alpha| \leq a(d_2+d_1)E_2/d_1$.

This event must be a strong reduction of type (iii) or (iv) because Equation (18) implies the latter holds when $Y_{C_i} \geq \alpha$, and its mirror image applies to the case $Y_{C_i} \leq \alpha$. Q.E.D.

Verification for the Case where $\ell_i < 2^{-s\sqrt{\log N}}$, $N_i > 2^{s\sqrt{\log N}}$, and $E_2 \leq (d_2/d_1)(E_1+E_3)$. With no loss of generality we may assume $\varepsilon_i^R \geq \varepsilon_i^L$. Equation (16) implies in this case that there is at least a probability 1/5 that $\alpha \leq Y_{C_i^{INT}} \leq \alpha + (E_1+E_2+E_3)$. This inequality, $\varepsilon_i^R \geq \varepsilon_i^L$, and Equation (17) then imply a strongly reducing event of type-(iv). Q.E.D.

Verification for the Case where $N_i \leq 2^{-s\sqrt{\log N}}$. Each 2-cycle will then automatically be strongly reducing because N_i satisfies reduction condition (i). Q.E.D.

Verification for $\ell_i \geq \ell/2^{S\sqrt{\log N}}$. The density u will always have an upper bound $b_1+b_2(b_4-b_3)$ and a lower bound b_1 over the interval $[b_3,b_4]$ by Equations (1) through (3). Probability theory [1,5] then certainly implies a binary cut with a probability > 1/2 will reduce ℓ_i by a factor of certainly $r=1-b_1/4[b_1+b_2(b_4-b_5)]$. Q.E.D.

Appendix Justifying Equation (15): This equation states E_1 and E_3 make positive contributions to $Y_{C_i^{INT}}$'s lower bound, and E_2's contribution is negative. The latter follows from a worst-case estimate of u's non-uniformity. E_1's contribution follows from the theory of order statistics [5], and E_3's contribution by applying $\varepsilon_i^R \geq \varepsilon_i^L$ to the algebraic properties of the rounding function in equation (7). The crucial part of equation (15) to our analysis is that it states E_1 and E_3 make contributions in the opposite direction as E_2's worst-case effect.

Further Results: Let a denote *any* fixed constant and N a variable denoting the initial size of the file. Define SPLIT(a) to be an iterative search procedure which uses C_i^{BIN} as the cut index when $\ell_i > a/\sqrt{N}$ and otherwise uses C_i^{INT}. A future paper [13] by Willard will show this algorithm has a complexity $O(\sqrt{\log N})$, in the sense that for each regular probability density u and parameter a there exists a constant K_{au} such that $K_{au}\sqrt{\log N}$ bounds the procedure's expected performance. This fact leads to an alternate proof of ALTERNATE's expected time $O(\sqrt{\log N})$ based on comparing the complexities of ALTERNATE and SPLIT(a). The proof methods in the two papers are quite different, and they should probably be read in conjunction. Also, see [11,12] for highly practical indexed data structures which produce log-logarithmic worst-case search times.

REFERENCES

1. W. Feller, An Introduction to Probability Theory and Its Applications, Vol. 1, third ed., Wiley, New York 1968.

2. M.L. Fredman, J. Komlos, and E. Szemeredi, "Storing a Sparse Table with O(1) Worst Case Access Time," Proc. Twenty-third Annual Symp. Foundations of Computer Science, 1982, pp. 165-170.

3. G.H. Gonnet, L.D. Rogers, and J.A. George, "An Algorithmic and Complexity Analysis of Interpolation Search," Acta Informatica, 13 (1980), pp. 39-52.

4. D.B. Johnson, "A Priority Queue in which Initialization and Queue Operations Take O(log log D) Time," Report No. CS 81-13, Penn. State University, July, 1981.

5. M.G. Kendall and A. Stuart, Advanced Theory of Statistics, Vol. 1, fourth ed., Macmillan, New York, 1977.

6. Y. Pearl, A. Itai, and A. Avni, "Interpolation Search - A Log Log N Search," Comm. ACM, 21 (1978), pp. 550-554.

7. Y. Pearl and E.M. Reingold, "Understanding the Complexity of Interpolation Search," Inform. Proc. Letters, 6 (1977), pp. 219-222.

8. A. Ralson and P. Rubinowitz, A First Course in Numerical Analysis, second ed., McGraw-Hill, New York, 1978.

9. P. Van Emde Boas, R. Kass, and E. Zijlstra, "Design and Implementation of an Efficient Priority Queue," Math. Systems Theory, 10 (1977), pp. 99-127.

10. D.E. Willard, "Searching Nonuniformly Generated Files in Log Log N Runtime," manuscript to appear in SIAM J. on Computing, has already appeared at Proceedings of 1981 ORSA-TIMS Conference on Applied Probability Computer Science Interface as extended abstract.

11. D.E. Willard, "Two Very Fast Trie Data Structures," The 19th Annual Allerton Conference on Communication, Control and Computing, (1981), pp. 355-363. Also to appear in JCSS.

12. D.E. Willard, "Log-logarithmic Worst-case Search Times Are Possible in O(N) Space," Inf. Proc. Letters, August 1983.

13. D.E. Willard, forthcoming SUNY-Albany report.

14. A.C. Yao and F.F. Yao, "The Complexity of Searching an Ordered Random Table," Proc. Seventeenth Annual Symp. Foundations of Computer Science, 1967, pp. 173-177.

ACKNOWLEDGEMENTS. I thank S. Bing Yao for suggesting I study the behavior of interpolation search on nonuniform densities, and I thank Jon Shapiro for suggesting I study how the log log N time from a previous paper by Willard [10] would change if the proof methods were applied to ALTERNATE. The answer to Jon's question is that ALTERNATE is less efficient than [10], but interesting because it is curious that the straightforward hybrid of two algorithms out performs its constituents asymptotically.

PREDICATE RETRIEVAL THEORY (EXTENDED ABSTRACT)

Dan E. Willard*
Bell Laboratories, and
Computer Science Department, State University of New York at Albany,
Albany, New York 12222

1. INTRODUCTION

We define a broad subset of the relational calculus and show all expressions in this language can be evaluated in space $O(N)$ and time $O(N \log^d N)$ for some small constant d whose value depends on the particular predicate. We estimate that 90% or more of all relational calculus expressions can be evaluated in time $O(N \log N)$ and memory $O(N)$ using this technique. Our system rests on applying the recent multi-dimensional and decomposable data structure theories to an unusual application where d can be made especially small and practical.

Overview of paper: The terms: "decomposition" and "decomposable searching" will often be used in this paper. They should not be confused and actually refer to entirely different concepts. The former refers to a process of splitting one relational language query into a sequence of subqueries. The latter views data structures as the primitive objects and attempts to perform more complicated queries by combining data structures, sometimes recursively. Examples of articles on relational language decomposition include [4 ,7-10,24, 27, 46, 52, 53, 56], and examples of articles on decomposable data structures are [12, 13, 22, 26, 46, 47, 51]. The data structures in the decomposable searching theory are sometimes defined recursively within it [6, 12, 35, 47, 51], other times stem from traditional search theory of the textbooks, and they may also come from multidimensional search theory [5 , 11, 19, 23, 34, 47, 48, 49, 55,58]. We combine the two approaches in this article, with Section 2 defining our language and reviewing the relational language decomposition literature, with Section 3 reviewing the relevant decomposable data structure theory, and with Section 4 synthesizing the two methods. Willard's dissertation [46] announced the time but not space complexities below.

2. STATEMENT OF THE QUERY PROBLEM

We define the language RCS in this section. This article will show how to execute all RCS expressions in time $O(N \log^d N)$ and space $O(N)$, for a constant d defined later. The acronym RCS stands for Relational Calculus Subset. All relational calculus expressions are not included in this language because that problem is NP - hard. RCS is an attempt to define a subset of the relational language which is simultaneously broad enough to include most of the requests likely to come from the commercial user and narrow enough to exclude the intractible expressions. To make RCS broad enough, we incorporate into this language calculus expressions that would not be considered as either tree or acyclic-based in the relational literature mentioned in the previous section.

Following the terminology in data base theory, the term relation will refer to a table that has j rows and m columns. A record corresponds to a row in this table and is often called a tuple. A particular column field of the record x is called an attribute and is denoted x.a. A data base schema is defined as a collection of several relations, and the terms Rx, Ry and Rz will denote the relations over which the record variable x, y and z span. |R| will denote the size of the relation R, i.e. its

---

* ONR grant N00014-76-C-0914 supported the 1978 draft [46] of this paper.

number of rows; the cardinality of the data base consisting of the relations $R_1 \ldots R_k$ will be defined as $\Sigma \mid R_i \mid$, *and it will be denoted as simply N*.

Each relation will be assumed to contain one attribute which uniquely identifies it. Codd calls this attribute a primary key, and its defining characteristic is that no two records store the same value in this field. For reasons explained at the end of this section, the expressive power of our language will be enhanced substantially if we divide all relations into two categories called *esets* and *associations* (between esets), following the terminology of the entity-relationship model [14]. Five types of atomic predicates will be used in this paper. These are defined below:

1) *Equality Predicates*: These are expressions of the form $x.a = y.b$ where Rx and Ry are relations of type eset.
2) *Order Predicates*: These are expressions of the form $x.a \geq y.b$ or $x.a > y.b$ where Rx and Ry are relations of the type eset.
3) *Unary Comparison Predicates*: These are expressions of one of the forms $x.a = c$, $x.a > c$, $x.a \geq c$, $x.a_1 = x.a_2$ or $x.a_1 > x.a_2$ where Rx is again an eset and c is a constant.
4) *Unary List Predicates*: These are expressions of the form: $\exists\, y \in Ry{:}\, y.a = x.prime$ where Rx is an eset and Ry an association.
5) *Tabular Predicates*: These are expressions of the form:
$$\{\exists\, z \in Rz{:}\ z.a = x.prime \;\&\; z.b = y.prime\} \tag{2.1}$$
where Rx and Ry are relations of type eset and Rz is an associative relation.

The symbols $u_1(x)$, $u_2(x)$ ....and $T_1(x,y)$, $T_2(x,y)$.... will denote unary list and tabular atomic predicates respectively; at the end of this section we will explain how these concepts and the distinction between associations and esets has major implications on performance.

An *unquantified normal* predicate will be defined as a logical expression which concatenates a series of atomic predicates with the logical connectives of AND, OR and NOT. For example *if* x, y, z and w are variables *ranging over esets* then equation (2.2) is an unquantified normal expression:
$$\{[x.a = y.b \;\&\; y.b > z.c] \text{ or NOT}[T(x,y) \;\&\; U(w)]\} \tag{2.2}$$

Logical quantifiers in our relational language will be denoted as Q_1, Q_2, etc. We use this term in a context somewhat more general than the traditional set-theoretic context. The quantifier $Q(x)$ will count the number of x in Rx that satisfies a specified condition and then apply a specified function which returns a boolean value. For instance, the quantifier $\exists x$ returns a value of TRUE iff COUNT $(x) \geq 1$, and the quantifier $\forall x$ will return TRUE when COUNT (x) equals the cardinality of Rx. A more novel quantifier might return TRUE when $5 <$ COUNT$(x) < 10$. Any function which requires $O(1)$ time to return a boolean value as a function of COUNT (x) will be considered a quantifier in the language RCS.

The final part of our calculus language will be the retreive-clause: This will be an expression of one of the forms:

i) $\text{FIND}(x_1, x_2 \ldots x_i)$
ii) $\text{COUNT}(x_1, x_2)$
iii) $\text{SUM}(x_1, x_2)$.

If $r(x_1\ x_2 \ldots x_i)$ is a retrieve clause, $j \geq i$, and $e(x_1, x_2 \ldots x_j)$ is a normal unquantified relational expression then a sentence of the form below will be said to be a *quantified calculus expression*
$$\{r(x_1\ x_2\ \ldots x_i)(Q_{i+1}(x_{i+1})\ \ldots Q(x_j){:}\ e(x_1\ x_2\ \ldots x_j)\} \tag{2.3}$$
The meaning of such an expression depends on the type of its retreive clause, and it is defined below:

i) If r is a clause of the type "FIND" then return all the i-tuples which satisfy the expression to the right of the retrieve clause and belong to the cross product $Rx_1 \times \ldots \times Rx_i$.

ii) If r is a clause of the type COUNT (x_1, x_2) then the calculus expression is an order to construct an array I( ) defined as follows: For each $x_1 \in Rx_1$, $I(x_1)$ is the number of different elements x_2 in Rx_2 satisfying the expression to the right of the retrieve clause.

iii) If r is a clause of the type SUM (x_1, x_2) and f is a pre-specified function that maps Rx_2 into a group, then the calculus expression is an order slightly more general than COUNT to construct an array $I^f($ $)$ defined as follows: For each $x_1 \in Rx_1$, $I^f(x_1)$ is the sum of $f(x_2)$ for the subset of Rx_2 satisfying the expression to the right of the retrieve clause. (Note: SUM and COUNT clauses always have two arguments.)

One additional comment on notation: Since the variables x, y, and z are always assumed to belong to the respective relations Rx, Ry and Rz, we do not formally write them into our relational calculus expressions. That is, our shorthand notations has (2.4) stand as an abbreviation for (2.5)

$$\{FIND(x)\ \forall y\ \exists z: e(x,y,z)\} \tag{2.4}$$

$$\{FIND(x \text{ in } Rx)\ \forall y \in Ry\ \exists z \in Rz: e(x,y,z)\} \tag{2.5}$$

Throughout this paper, q will denote a normal relational calculus expression. The graph of this expression, denoted G(q), will be defined to have vertex set corresponding to the eset variables appearing in q and a directed edge from x to y iff these two vertices appear together in some atomic predicate and the retrieve or quantifier clause defining x lies left of y's definition in q. For instance, the graph of equation (2.6) is a tree.

$$\{FIND(x,y)\ \forall z\ \exists w: x.a > w.b\ \&\ T(x,y)\ \&\ y.b = z.c\} \tag{2.6}$$

RCS is defined as the subset of the normal relational calculus whose graph is a tree or forest. Equation (2.6) is therefore an example of a member of RCS. We will now provide some intuition for why it would be interesting to develop an algorithm for performing all RCS queries in $O(N \log^d N)$ time and O(N) space.

The query class would not be adequately broad if we excluded the notion of associative relations and therefore also excluded tabular predicates. For instance, if Rx, Ry and Rz are each relations of type eset then equation (2.7) would not belong to RCS because its graph is not a tree.

$$\{FIND(x,y)\ \exists z: x.a > y.b\ \&\ x.prime = z.a\ \&\ y.prime = z.b\} \tag{2.7}$$

However, if one thinks of Rz as an associate relation and if T(x,y) denotes the tabular predicate in equation (2.1) then (2.7) is clearly equivalent to the elements satisfying:

$$\{FIND(x,y): x.a > y.b\ \&\ T(x,y)\} \tag{2.8}$$

The first point is that the latter equation is an RCS expression and this formalization is what we need to describe those expressions which at first appear to be cyclic but can nevertheless be processed efficiently. The number of predicates added to our query language after introducing the notion of tabular predicates based on associative relational is considerable. We estimate that a good many of the textbook examples of relational queries can be converted into RCS expressions. Presumably, this language also models the typical needs of a commercial user.

The second although lesser advantage of associative relations is that they make the data base language more accessible to a casual user.

For instance, Equation (2.8) is much easier to read than (2.7). This point about the user-friendly nature of entity-relationship primitives has appeared elsewhere in the literature [1, 14, 28, 30-33 , 42, 43], and it will not be discussed further here except to state that there is no disadvantage to dividing relations into the categories of esets and associations from a performance point of view.

A relation could belong to one category in one query and to the opposing category in the next query. The notion of associations and tabular predicates simply offers a shorthand mechanism for expressing the very broad class of queries we can process in good time and space.

Let RCS-zero denote the subset of the language RCS whose expressions have no order predicates. Now we will offer a brief review of the relational language decomposition literature. Theorem 7.5L of Willard's dissertation [46] announced that each RCS query had a constant d such that it could be processed in $O(N \log^d N)$ time and space, and Theorem 7.5M indicated the time and space was $O(N)$ for the special case of RCS-zero expressions. The dissertation announced Theorem 7.5L could be proven by decomposing RCS queries into a sequence of subroutine calls to the earlier chapters of Willard's dissertation, but it omitted giving the formal proof until Willard developed a yet stronger method reducing the memory cost of all RCS queries to $O(N)$. We have done so now, using a new technique from Edelsbrenner and Overmars [22]. The potential commercial significance of the rest of this paper will be apparent to readers familiar with the literature on acyclic data bases [4 , 7 , 8 , 9 , 10, 15, 20, 24, 27, 29, 32 , 46, 52, 53, 56, 57]; essentially our results for both RCS and RCS-zero are a major improvement in performance beyond those aspects of these papers related to efficiency.

3. REVIEW OF DECOMPOSABLE DATA STRUCTURES

All measurements of complexity in this paper will be done in terms of a quantity called worst-case hashing time. This measure is worst case in every sense except it assumes average luck when hashing. If a query requires time $A+j$ and space $B+j$ to produce and to store a list of j tuples, we will say this algorithm runs in locate time $O(A)$ and locate space $O(B)$. The symbol j is omitted from the locate notation because it is an unavoidable cost when producing a list of size j, as Willard has explained in [49]. If the query asks to produce an aggregate, we will not say it runs in a time-space combination (A,B) unless this is literally its cost. Locate terminology should be avoided for the latter problem because one can normally produce an aggregate describing j elements without processing the elements one by one. An algorithm is said to run in quasi-linear time and linear space iff $A = N \log^d N$ and $B = N$.

An E-8 predicate $e(x,y)$ is an arbitrary unquantified predicate expression whose equality, order, tabular and unary predicates discuss the two variables x and y. There are precisely four kinds of relational calculus expressions that RCS allows us to build around the class of two-variable predicates; these are listed below and called E-8 queries:

i) $\{\text{FIND}(x)Q(y): e(x,y)\}$ (3.1)
ii) $\{\text{FIND}(x,y): e(x,y)\}$ (3.2)
iii) $\{\text{COUNT}\ (x,y): e(x,y)\}$ (3.3)
iv) $\{\text{SUM}(x,y): e(x,y)\}$ (3.4)

Henceforth, $D(e)$ will denote the number of distinct y-attributes that can be found in $e(x,y)$'s order predicate comparisons with x, and $\delta(e)$ will equal 1 when $D(e) \le 1$ and zero otherwise. The full-length version of this paper proves retrievals of types (3.1), (3.3) and (3.4) run in space $O(N)$ and time $O\ \mathrm{Max}(N \log^{D(e)-1} N;\ \delta(e)\ N \log^{D(e)} N)$, and most retrievals of type (3.2) also are bounded by this time, with the exceptions consuming time $O(N \log^{D(e)} N)$. These quantities are usually very small numbers; for instance $D(e_1) = 2$, $D(e_2) = 0$, $D(e_3) = 0$, and $D(e_4) = 1$ in the equations below:

$$e_1 = \{y.b_1 > x.a_1 \ \&\ y.b_2 \neq x.a_2 \ \&\ y.b_3 > x.a_3\}$$

$$e_2 = \{[y.b_1 = x.a_1 \ \&\ T.(x,y)] \text{ or } y.b_2 = x.a_2\}$$

$$e_3 = \{y.b_1 > c \ \&\ u(x) \ \&\ y.b_1 = x.a_1\}$$

$$e_4 = \{y.b_1 > x.a_1 \ \&\ y.b_1 > x.a_2 \ \&\ y.b_1 < x.a_3\}$$

The proposition above is called the Batch E-8 Theorem. Its proof [50] is literally 40 pages long and a condensation of a more elaborate 250 page proof of a stronger dynamic result. I would gladly mail the short proof to any one who writes me at SUNY Albany, and the longer proof is published in [46]. The next few paragraphs of this abstract will provide at least enough background information so the reader can gain a partial understanding of the nature of the Batch E-8 Theorem.

The term augmented tree refers to a data structure discussed in [6, 12, 13, 21, 22, 26, 34, 46, 47, 51], which has two parts. Its first section is called the base and consists of a binary tree where the records are stored in some specified order. Henceforth, we will call an element of Ry a y-record. A possible base would be a tree that stores the y-records in order of increasing attribute value $y.b_2$. The second part of an augmented tree is its set of auxillary fields. For each internal node v of the base, AUX(v) will consist of some data structure describing v's descendants. An example would be AUX(v) fields that enumerate the y-records descending from v in order of increasing $y.b_1$ value. See [47] for both a good example of an augmented tree and a surprising observation about their efficiency.

Willard's dissertation advocated using literally a different augmented tree for each different E-8 batch query. These operations would require the allocation of $O(N \log^{D(e)} N)$ time and space for each different batch query. An observation made later in [47] would reduce some of these complexities by a factor log N, but we still hesitated to publish Theorem 7.5L because its memory space exceeded the benchmark O(N) which separated commercial data base applications from academic data structures. Recently Edelsbrenner and Overmars [22] announced that the memory space of essentially all augmented trees could be reduced to O(N) at no loss in time for the special case where N queries are performed in batch! The essence of their proposal is that only one of the augmented tree's fields should be constructed at a particular moment in time, and all the queries intending to search that field should do so precisely when it is constructed. This method, called memory streaming, was used previously in the Bentley-Shamos ECDF batch algorithm, but Edelsbrenner and Overmars were the first to propose it as a general algorithmic technique.

Memory streaming changes the significance of Chapters 4 through 6 of Willard's dissertation radically. It reduces the memory use of Willard's batch data structures from quasi-linear to linear in N. The second-generation d-fold tree [47] reduces many of [46]'s time complexities by a factor log N. The resulting complexities were presented at the beginning of this section, and the reader should write Willard for [50] if he wishes to see the Batch E-8 Algorithm. The next section will explain the algorithm's implications for data bases. The Appendix provides an example of a batch E-8 operation, but it is not sufficient to understand the breadth of the Batch E-8 theory.

4. THE PROOF OF THEOREM 7.5L FROM WILLARD'S DISSERTATION AND GENERAL DECOMPOSITION THEORY

Definition 4.1. Two RCS queries are called equivalent iff the answers produced by both queries are necessarily the same. Sometimes, the second query among a pair of equivalent RCS expressions will be constructed by executing a sequence of E-8 queries, and then using the result to create some unary list predicates appearing in the second query. In that case, the second query will be called an E-8 reduction of the first.

Example 4.2: Let e_1, e_2, and e_3 denote three E-8 expressions and consider the RCS query

$$\{\mathrm{FIND}(x,y)\ \forall\ z\ \exists\ w\colon e_1(x,y)\ \&\ e_2(x,z)\ \&\ e_3(y,w)\}. \tag{4.5}$$

Let L_2 and L_3 denote the subsets of Rx and Ry satisfying the E-8 queries

$$L_2 = \{\mathrm{FIND}(x)\ \forall\ z\colon e_2(x,z)\} \tag{4.6}$$

$$L_3 = \{\mathrm{FIND}(y)\ \exists\ w\colon e_3(y,w)\} \tag{4.7}$$

Then if $u_2(x)$ and $u_3(y)$ denote the unary-list predicates associated with the lists L_2 and L_3, equation (4.8) is an example of an "E-8 reduction" whose solution set is equivalent to the original query in (4.5).

$$\{\mathrm{FIND}(x,y)\colon e_1(x,y)\ \text{and}\ u_2(x)\ \text{and}\ u_3(y)\}. \tag{4.8}$$

One would construct the query in (4.8) by using the Batch E-8 algorithm to execute the queries on the right sides of (4.6) and (4.7), and then using the result to build the unary predicates $u_2(x)$ and $u_3(y)$. Naturally, our reason for decomposing an RCS query into E-8 queries is that the algorithms for executing the latter are very efficient, by the Batch E-8 Theorem.

Sometimes the relational graph of an RCS expression will form a forest rather than a tree. In that case, our discussion will be simplified if we add a few "pseudo-edges" to the relational graph so that it becomes a tree. These pseudo-edges will be defined to go from the root of each tree in the original forest to the leftmost variable declared in the query statement. The resulting modified relational graph is called an induced relational tree.

Lemma 4.3: For each $0 \le j' < j$, every RCS query with j quantifiers can be E-8 reduced to an equivalent query with j' quantifiers.

Proof: We prove the theorem for the case where $j' = j-1$. The proof for other values of j' follows by an easy inductive argument where one repeats the process described below a total of (j-j') times.

Let equation (4.9) denote the initial RCS query:

$$\{(\mathrm{retrieve}\ (x_1x_2\ldots x_i)Q_{i+1}(x_{i+1})\ldots Q_{i+j}(x_{i+j})\colon e(x_1x_2\ldots x_{i+j})\} \tag{4.9}$$

Let x_f denote the father of x_{i+j} in the induced relational tree of the query and $e_1^Ae_2^A\ldots e_k^A$ and $e_1^Be_2^B\ldots e_k^B$ a series of unquantified normal predicates such that

i) the predicates $e_1^Ae_2^A\ldots e_k^A$ are disjoint and each is defined over the variable set $x_1x_2\ldots x_{i+j-1}$.

ii) The predicates $e_1^Be_2^B\ldots e_k^B$ contain atomic predicates that only use the variables x_{i+j} and x_f.

iii) These predicates satisfy the condition

$$e = \{(e_1^A\ \&\ e_1^B)\ \text{or}\ldots\text{or}\ (e_m^B\ \&\ e_m^B)\} \tag{4.10}$$

Let L_m denote the subset of Rx_f produced by the following E-8 queries:

$$L_m = \{FIND(x_f)Q_{i+j}(x_{i+j});\ e^B_m(x_f,x_{i+j})\} \qquad (4.11)$$

Then if $u_m(x_f)$ is the unary predicate induced by Lm, let $e^*(x_1x_2\ldots x_{i+j-1})$ denote the predicate

$$e^* = \{(e^A_1 \ \&\ u_1(x_f)) \ \&\ \ldots \ \&\ (e^A_m \ \&\ u_m(x_f))\}.$$

From the last three equations, it is evident that an E-8 reduction which removes one of the quantifiers from (4.5) but produces an equivalent expression is:

$$\{retrieve(x_1\ldots x_i)Q_{i+1}(x_{i+1})\ldots Q_{i+j-1}(x_{u+j-1}):\ e^*(x_1\ldots x_{u+j-1})\} \qquad (4.13)$$

Q.E.D.

Corollary 4.3: Every RCS query whose retrieve clause has only one or two variables can be evaluated by executing a sequence of E-8 queries.
Proof: Immediate consequence of Lemma 4.3.

Definition 4.4: A RCS query is a complex join iff its retrieve clause is of the type FIND and it has no logical quantifiers; i.e., it is an expression of the form:

$$\{FIND(x_1,x_2,\ldots,x_i):\ e(x_1,x_2,\ldots,x_i)\}. \qquad (4.14)$$

Corollary 4.5: Every RCS expression which has three or more variables in its retrieve clause can be E-8 reduced to a complex join.
Proof: Another easy consequence of Lemma 4.3.

Define an atomic term to be either an atomic predicate or a term of the form "NOT A" where A is an atomic predicate. A c-complex join will be defined to be a complex join where the term $e(x_1\ldots x_i)$ in equation (4.14) is a conjunction of atomic terms. Most of the rest of this section will study c-complex joins and show how the E-8 algorithms will reduce them to an easily solvable problem.

Let T denote the induced relational tree of the c-complex join and σ an edge of the tree. The edge σ will also be associated with a permutation $\{\sigma(j)\}$ of the relations $Rx_1Rx_2\ldots Rx_i$ such that if σ is an edge from x_g to x_h then

i) $\sigma(1) = h$, $\sigma(2) = g$ and $\sigma(j)$ for $3 \le j \le i$ lists the remaining integers between 1 and i, and

ii) the permuted query below belongs to the class RCS.

$$q_\sigma = \{FIND(x_{\sigma(1)},x_{\sigma(2)})\ \exists x_{\sigma(3)} \ \ldots\ \exists x_{\sigma(i)}:\ e(x_1\ldots x_i)\} \qquad (4.15)$$

It is trivial to verify that if the original query (4.14) was a c-complex join then each edge σ has at least one permutation $\sigma(j)$ such that q_σ is also an RCS query. In our discussion S_σ will denote the subset of $Rx_{\sigma(1)} \times Rx_{\sigma(2)}$ that is retrieved by (4.15).

Lemma 4.7: Let q be a c-complex join query, T its relational tree, and σ^* that edge of T where S_{σ^*} has the largest cardinality. Then in a conventional (as opposed to locate) cost model, the time and memory for constructing all the sets S_σ will be $\le K(|S_{\sigma^*}| + N \log^d N)$ and $K(|S_{\sigma^*}| + N)$, respectively, for some constants K and d whose values depend only on q.
Proof: The canonical form of a query of the type q_σ is shown in equation (4.15). By Section 3 and Corollary 4.3, these queries will produce the S_σ in time $O(|S_\sigma| + N \log^{d(\sigma)} N)$ and space $O(|S_\sigma| + N)$ under conventional cost models. Let K_σ be an upper bound on the coefficients hidden inside

the O-notation, $K = \Sigma_{\sigma \in T} K_\sigma$, and $d = \text{MAX}(d(\sigma) \mid \sigma \in T)$. Then clearly the time and space to produce all the sets of form S_σ are bounded by $K[|S_{\sigma*}| + N \log^d N]$ and $K(|S_{\sigma*}| + N)$, respectively. Q.E.D.

Let (, , , ,) denote an ordered i-tuple and S the following subset of the cross product space $Rx_1 \times Rx_2 \times \ldots \times Rx_i$;

$$S = \{(x_1,x_2,\ldots,x_i) \mid \forall\sigma \in T: (x_{\sigma(1)},x_{\sigma(2)}) \in S_\sigma\}. \quad (4.16)$$

In the terminology of the relational algebra, equation (4.16) is called a "natural equijoin" expression [45]. The term S_σ are called the input relations, and their purpose is to define a new product relation S. The problem of efficiently constructing the product relation S in a natural equijoin has been studied very extensively [2, 4, 7-10, 15,27,31,36,45,56].

In general, there is no algorithm that guarantees all natural equijoins run efficiently. One problem is that the product join could potentially have size $|Rx_1| \cdot |Rx_2| \ldots |Rx_u|$, and its construction could therefore exhaust all time and space. Relational algebracists have overcome this particular problem by looking at natural joins from the same perspective as data structure theorists. That is, the time and space costs are written as terms of the form $O(f + |S|)$, and one seeks to minimize only the term f, i.e., optimize "locate" rather than "locate-and-enumerate" costs.

Even in this reformulated version, the general natural join problem is NP-hard, However, some progress has been made using the notion of semijoin developed originally by Palermo [40].

If a join has input relations S_σ and product relation S, then S_σ is called semijoin reduced if for each $z \in S_\sigma$ there is at least one element $z^* \in S$ such that Z's components match z*'s on their overlapping domains. An axiom of join theory is that if all the S_σ are semijoin reduced then the join is cheap, and its expected hashing time and memory costs can actually be made O(1) in the locate terminology, by essentially a trivial algorithm (see for instance [27, 45]. The problem is that the input relations are generally not initially semijoin reduced, and it is expensive to bring them to such a state. A key problem is to control this cost.

Now here is the first main point: The input relations in our particular application are defined by equation (4.15), and the existential quantifiers in that equation guarantee the S_σ are semijoin reduced. This means after one has constructed the S_σ (a potentially formidable operation), the building of S is cheap, i.e.,

Lemma 4.8: The join in equation (4.16) has a locate hashing time and memory cost O(1).

The second main point is that if the original query q is a c-complex join then the construction of S_σ is also inexpensive, by Lemma 4.7.

Collecting these two results we now prove:

Theorem 4.9: c-Complex join queries run in quasi-linear locate time and linear-locate space.

Proof: The algorithm for executing these queries has two steps. First it constructs the sets S_σ by using the first half of this section to decompose the S_σ into an E-8 execution sequence, and then it applies the results in the Batch E-8 Theorem to execute these queries. Since $|S_{\sigma*}| \le S$, Lemma 4.7 states this first step meets the claimed time and space constraint! The second step takes the natural join of the S_σ, and Lemma 4.8 certainly shows its costs satisfy the Theorem 4.7. Q.E.D.

The time complexities in the next theorem were announced in section 7.5 of Willard's dissertation [46]. The memory complexities are new because of our new theorem from Section 3.

Theorem 4.10: All RCS queries run in quasi-linear time and in linear space.

Proof: Corollary 4.3 already proved Proposition 4.10 for the queries with one and two variable retrieve clauses. For larger queries, consider the following procedure:

1) Apply the procedure from Corollary 4.5 to remove all the quantifiers from the RCS query.
2) Find a sequence of c-complex predicates satisfying $e = \{e_1 \vee e_2 \; \ldots \; e_k\}$. Apply the procedure from Theorem 4.9 to find the sets of tuples $S_1 S_2 \ldots S_k$ that satisfy these complex joins. The union of these sets is the answer to the user's query.

The combination of Lemma 4.1, Corollary 4.5, Theorem 4.9, and the Batch E-8 Theorem imply that this procedure meets the performance claims of Theorem 4.10. Q.E.D.

Remark 4.11: The algorithms in this section were defined very carefully. Slightly different algorithms would appear to be similar but actually are much less efficient. For example, let $e(x_1 x_2,,,x_i)$ denote the predicate in (4.14), and assume it is a c-complex join. For each edge σ, define $e_\sigma(x_{\sigma(1)}, x_{\sigma(2)})$ to be a c-complex predicate whose terms are those terms from e that contain only the variables $x_{\sigma(1)}$ and $x_{\sigma(2)}$. Define $Z_\sigma = \{\text{FIND}(x_{\sigma(1)} x_{\sigma(2)}): e_\sigma\}$. Then an alternate procedure for finding a c-complex join would be to calculate the Z_σ and apply any one of the algorithms from [4,7,9, 27, 53, 56] to derive their natural equijoin, denoted as S. This method is much less efficient than Theorem 4.9 because the intermediate relations Z_σ are not guarantted to satisfy $|Z_\sigma| < |S|$. That is, the algorithm would exceed good locate time if any $|Z_\sigma| >> |S|$.

APPENDIX TO SECTION 3

First we will consider how to construct in time O(N) the array I(x) that is the answer to the RCS query {COUNT(x,y): e(x,y)} for the predicate e defined below:

$$e(x,y) = [x.a_1 = y.b_1 \text{ or } (x.a_2 = y.b_2 \;\&\; x.a_3 \neq y.b_3)] \tag{A.1}$$

Let $I_j(c)$ denote the number of y-records satisfying $y.b_j = c$, $I_{jk}(c_1,c_2)$ the number of y-records satisfying the conjunction $(y.b_j = c_1 \;\&\; y.b_k = c_2)$, and $I_{jkm}(c_1,c_2,c_3)$ the similar concept for ordered triples. A construction of $I_j(c)$ is defined as a hashing function whose hash keys are terms of the form c and which stores a record containing the integer $I_j(c)$ iff this quantity is non-zero. It is easily proven one can produce such a hashing construction in O(N) time, and the result generalizes easily for I_{jk} and I_{jkm}. Then we can produce in roughly 5N time the hash constructions I_1, I_2, I_{12}, I_{23} and I_{123}; and in 5N further time we can produce the array I(x) by scanning through R_x and using the formula below:

$$I(x) = I_1(x.a_1) + I_2(x.a_2) - I_{12}(x.a_1,x.a_2) - I_{23}(x.a_2,x.a_3) + I_{123}(x.a_1.x.a_2,x.a_3)$$

Next suppose the E-8 query is of the form of equation (A.2) or (A.3)

$$\{\mathrm{FIND}(x)\ \exists\ y: e(x,y)\} \tag{A.2}$$

$$\{\mathrm{FIND}(x)\ \forall\ y: e(x,y)\} \tag{A.3}$$

The first query may be solved in time $O(N)$ by using the previous paragraph to calculate the array $I(x)$, and then listing those particular $\overline{x}$ satisfying $I(\overline{x}) \geq 1$. The evaluation of Equation (A.3) is the same except we list those $\overline{x}$ satisfying $I(x) = |Ry|$.

I like this particular example because it uses time $O(N)$ for a task which required time $O(N^2)$ in the previous data base literature. An example very similar to the last three equations was the centerpiece of my 1976 memo describing a proposed dissertation topic to Ugo Gagliardi. To be sure, there are many other facets of data base optimization related to augmented trees (Section 3), decomposition (Section 4), and the especially difficult aspects of optimizing equation (3.2) needed to establish Theorem 4.10. But we can't discuss all topics from a dissertation in a ten page abstract, and this example from my 1976 thesis proposal explains why I have been so obsessed with range query questions for the last seven years.

REFERENCES

1) J. Abrial, "Data Semantics" Data Base Management, North Holland Publishing Co., 1974, pp 1-60.
2) A. Aho, C. Berri and J. Ullman, "The Theory of Joins in Relational Databases" ACM Transactions on Database Systems (4), 1979, pp 297-314.
3) A. Aho, Hopcroft, J.E., and Ullman, J.D.,The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, Mass., 1974.
4) P.A. Bernstein and D.W. Chiu, "Using semijoins to solve relational queries," J.ACM 28 (Jan. 1981), pp 25-40.
5) J.L. Bentley, "Multidimensional Binary Search Trees using for Associative Searching," CACM 18 (1975), pp 509-517.
6) J.L. Bentley, "Multidimensional Divide-and-Conquer," Comm. of ACM 23 (1980),pp 214-228.
7) C. Beeri, et al., JACM 30 (1983), pp 479-514.
8) C. Berri, R. Fagin, D. Maier, A. Mendelzon, J. Ullman and M. Yannakakis, "Properties of Acyclic Data Base Schema," 13-th ACM Symp. on Theory of Computing, 1981, pp 355-362.
9) P. A. Bernstein and N. Goodman, "The Power of Natural Semijoins," SIAM J. Comput. 10(1981), pp 751-771.
10) P.A. Bernstein and N. Goodman, "The Power of Inequality Semijoins," Inf. Syst. (6) 1981, pp 255-264.
11) J.L. Bentley and H.A. Maurer, "Efficient Worst-Case Data Structures for Range Searching," Acta Information 13(1980), pp 155-168.
12) J.L. Bentley and M.I. Shamos, "A Problem in Multivariate Statistics: Algorithm, Data Structure, and Applications," Proc. of the 15-th Allerton Conf. on Communication, Control and Computing (1977), pp 193-201.
13) J.L. Bentley and J.B. Saxe, "Decomposable Searching Problems #1: Static to Dynamic Transformations," J. of Algorithms 1 (1980), pp 301-358.
14) T.P.S. Chen, "Entity Relationship Model: Toward A Unified View of Data", TODS (1), 1976, pp 9-36.
15) D.M. Chiu and Y.C. Ho, "A methodology for interpreting tree queries into optimal semijoin expressions", Proc. ACM Sigmod Int. Conf. Management of Data Bases, New York, 1980, pp 160-178.

16) E.F. Codd, "A relational model for data for large shared data banks", Comm. of the ACM 13, (1970), pp 377-397.
17) E.F. Codd, "A data base sublanguage founded on the relational calculus" Proc. 1971 ACM-SIGFIDET, pp 35-68.
18) E.F. Codd, "Relational algebra", Courant Computer Science Symposia 6, "Data Base Systems", New York, May 1971, Prentice-Hall, N.Y., 1971.
19) R. Cole and C. Yap, paper to appear at 1983 FOCS.
20) U. Dayal, "Processing queries with quantifiers: A horticultural approach", 2nd SIGACT-SIGMOD Symp. on Principle of Database Systems, 1983, pp 125-137.
21) H. Edelsbrenner, "A Note on Dynamic Range Searching", Bulletin of EATCS (15), 1981, pp 34-40.
22) H. Edelsbrenner and M. Overmars, "Batch Solutions to Decomposable Search Problems", 1983, U. Utrecht TM RVV-CS-83-8.
23) H. Edelsbrenner and E. Welzl, "Halfplans Range Search in Linear Space and $O(n^{.695})$ Query Time", Univ. of Graz Report F-111, 1983.
24) R. Fagin, JACM 30(1983), pp 514-551.
25) M. Fredman, J. Komlos and E. Szemeredi, "Storing a Space Table with O(1) Worst-Case Access Time, 23rd IEEE Symp. on Foundation of Comp. Sci., 1982, pp 165-170.
26) M.F. Fredman, "A Lower Bound on the Complexity of Orthogonal Range Queries", Journal of ACM, 28 (1981), pp 696-706.
27) N. Goodman and O. Shmueli, "Tree Queries: A simple class of Relational Queries", TODS 4 (1982), pp 653-678.
28) M. Hecht and L. Kerschberg, "Update Semantics for the functional data base model", Bell Labs technical memorandum 1980. Write Kershberg at Univ. of South Carolina to get a copy.
29) M. Jarke and J. Koch, "Range Nesting: A fast way to evaluate quantified queries", ACM SIGMOD Symp., (1983), pp 196-206.
30) S. Koenig and R. Paige, "A Transformational Framework for the Automatic Control of Derived Data", 7th Int. Conf. on VLDG, 1981, pp 306-318.
31) L. Kerchberg, P.D. Ting and S.B. Yao, "Query Optimization in Star Computer Networks", ACM Trans. on Data Bases (4) 1982, pp 653-677.
32) Y.E. Lien, "On the equivalence of Data Base Models", JACM 29 (1982), pp 333-362.
33) Y.E. Lien, J. Shapiro and S. Tsur, "DSIS: An Experimental Database System", 7th Int. Conf. on VLDB, 1981, pp 306-318.
34) D.T. Lee and C.K. Wong, "Worst Case Analysis for Region and Partial Region Searches in Multidimensional Binary Search Trees and Balanced Quad Trees", Acta Inf. 9 (1977), pp 23-29.
35) D.T. Lee and C.K. Wong, "Quintary Tree: A File Structure for Multidimensional Database Systems", ACM Trans. on Database Systems 5 (1980) pp 23-29.
36) D. Maier, Y. Sagiv and M. Yannakakis, "On the complexity of testing implications of functional and join dependencies", J. ACM 28, 4 (Oct. 1982), pp 680-695.
37) D. Maier and J.D. Ullman, "Connections in acyclic hypergraphs", Proc. ACM Symp. Principles of Database Systems, ACM, N.Y., 1982, pp 34-39.
38) M. Overmars and J. von Leeuvwen, "Dynamics Multi-Dimensional Data Structures Based on Quad and K-d Trees", Acta Inf. (17), 1982, pp 267-286.
39) R. Paige's dissertation, N.Y.U., 1979, available from Rutgers Univ.
40) J. Palermo, "A database search problem", 4-th Symp. on Comp. Inf. Syst., (COINS IV), 1972.

41) D.J. Rosenkrantz, R.E. Stearns and P.M. Lewis, "Systems level concurrency control for distributed database systems", ACM Trans. Database Syst. 3, (1978), pp 178-198.
42) D. Shipman, "The Function Data Model and the Data Language Daplex", TODS 6 (1981), pp 140-173.
43) E.H. Sibley and L. Kerschberg, "Data Architecture and Data Model Consideration", Proc. NCC, AFIPS, 1977, pp 85-96.
44) R.E. Stearns, P.M. Lewis II, D.J. Rosenkrantz, "Concu rency controls for database systems", Proc. 17th Ann. Symp. on Foundations of Computer Science, IEEE, N.Y., 1976, pp 19-32.
45) J.D. Ullman, Principles of Database Systems, Computer Science Press, Potomac, MD., 1982.
46) D.E. Willard, Predicate-Oriented Database Search Algorithms, Ph.D. thesis, Harvard Univ., 1978. Also in Oustanding Dissertations in Computer Science, Garland Pub., N.Y., 1979. The Garland copies are priced as hard-cover books; you can save money by asking your librarian to purchase a copy.
47) D.E. Willard, "New Data Structure for Orthogonal Queries", first draft was Harvard TR-22-78 (1978), second draft appeared in Proc. 1982 Allerton Conf., third draft to appear in SIAM J. Comp.
48) D.E. Willard, "Balanced Forests of K-d* Trees as a Dynamic Data Structure", Technical Report TR-23-78 (1978), Center for Research in Computing Technology, Harvard Univ., 1978. Revised and reissued as "K-d Trees in a Dynamic Environment", Technical Report No. 80-1, Dept. of Comp. Sci., Univ. of Iowa, 1980. To appear in CACM.
49) D.E. Willard, "Polygon Retrieval", SIAM J. on Comp., (11), 1982, pp 149-162.
50) D.E. Willard, "Abstract Predicate Retrieval Theory, SUNY Albany Technical Report 83-3, 1983.
51) D.E. Willard and G.S. Lueker, "Adding Range Restriction Capability to Dynamic Data Structures", to appear in J. of ACM.
52) E. Wong and K. Youssefi, "Decomposition - a strategy for query processing", ACM Trans. Database Syst., 1 (1976), pp 233-241.
53) M. Yannakakis, "Algorithms for acyclic database schemas", 6-th Int. Conf. on VLDB (1981) pp 82-94.
54) A.C. Yao, "Space-Time Tradeoff for Answering Range Queries", 14 ACM Conf. on Theory of Computing, 1982, pp 128-135.
55) F. Yao, "A 3 space partition and its applications", 15th ACM Conf. on Theory of Computing, 1983, pp 258-264.
56) C.T. Yu and M.Z. Ozsoyoglu, "An algorithm for tree-query membership of a distributed query", Compsac 79, IEEE Comp. Soc., 1979, pp 306-12.
57) C. Zaniolo, Analysis and design of relational schemata for database systems, Ph.D. dissertation, Tech. Rep. UCLA-Eng. 7769, Dept. Comp. Sci., Univ. California at Los Angeles, July 1976. This was the first paper in the literature discussing data base hypergraphs.
58) B. Chazelle, forthcoming paper at FOCS, 1983.

ACKNOWLEDGEMENTS

I warmly thank my thesis advisor, Prof. Ugo Gagliardi. Aside from arranging a joint Ph.D. program from Harvard's Math and Computer Science Departments, Ugo Gagliardi made the timely suggestion in 1975 that I study relational calculus optimization theory.

I thank several of my computer science friends: Eugene Cramer, Sam Kashdan, Larry Kerschberg, Robert Paige and his wife Nieba, Shlomo (Dick) Tsur, and Carlo Zaniolo for urging me to persist in researching data bases. I thank Jon Bentley for passing me some literature. I thank my Bell Labs management, B. Gopinath, Ed Lien and Eric Wolman for their patient support.

SOME PROPERTIES OF MULTI-ATTRIBUTE FILE SYSTEMS BASED UPON MULTIPLE KEY HASHING FUNCTIONS

C. C. CHANG and D. H. Su

Institute of Applied Mathemtics
National Chung Hsing University
Taichung, Taiwan, Republic of China

ABSTRACT

Rothnie and Lozano suggested the multiple key hashing (MKH for short) method. Lin, Lee and Du later showed that the file structure suggested by Rothnie and Lozano was a good file structure because they cluster similar records together. We assume that attributes in each partial match query are independently specified throughout this paper. We derive the performance formula of multi-attribute file systems based upon MKH functions. We also provide a method of designing theoretically optimal multi-attribute files based upon MKH functions. We further show that the problem of designing optimal multi-attribute files based upon MKH can be reduced to finding minimal N-tuples which was discussed by Chang, Lee and Du. Feasible approaches of designing optimal files based upon MKH under some limited conditions will be found.

SECTION 1. INTRODUCTION

Recently much attention has been paid to designing good multi-attribute file systems for partial match queries [Aho and Ullman 1979, Bentley and Friedman 1979, Bolour 1979, Chang 1982, Chang 1983, Chang, Du and Lee 1981, Chang and Fu 1978, Chang and Lee 1979, Chang, Lee and Du 1980, Chang, Lee and Du 1982, Ghosh 1977, Lee 1981, Lee and Tseng 1979, Lin, Lee and Du 1979, Liou and Yao 1977, Rivest 1976, Rothnie and Lozano 1974]. By a multi-attribute file system, we mean a file system containing a collection of records, where each record is characterized by a list of attributes. By partial match queries, we mean queries of the form $(A_1=a_1, A_2=a_2, \ldots, A_N=a_N)$, where for $1 \le i \le N$, a_i is either a key belonging to D_i, the domain of the i-th attribute, or is unspecified (i.e., a don't care condition) in which case it is denoted by * and where the number of unspecified attributes is j where $1 \le j \le N-1$. A response to a partial match query is a list of all records in which the i-th attribute is equal to a_i if a_i is not a *.

In this paper, we shall assume that partial match queries are not random. The probability that a particular attribute is specified in a query depends on the attribute, but will be assumed independent of what other attributes specified by the query.

The problem of multi-attribute file design can be explained by the following example.

Example 1.1:

Suppose we have a file with each record containing of three attributes: age, department code and English grade. Suppose these three attributes are attached with hypothetical probabilities as shown in Table 1.1.

Attribute	Probabilities
Age	0.7
Department code	0.2
English grade	0.8

Table 1.1. Attributes and Probabilities.

In the formulation the probability of every particular form of query can be found by multiplying the appropriate probabilities of the associated attributes. For example, in the case we see that about 45 percent of all queries specify age and English grade and nothing else. That is $0.7 \times (1-0.2) \times 0.8 = 0.448$. The probability of the query: List all students whose department code is "AM" is $0.3 \times 0.2 \times 0.2 = 0.012 = 1.2\%$.

Thus the problem of multi-attribute file system design for partial match queries can be stated as follows: Given a set of N-attribute records, a fixed number of buckets (NB), and N hypothetical probabilities attached to N attributes respectively, our goal is to arrange the records into NB buckets in such a way that the average number of buckets need to be searched over all possible partial match queries is minimized.

Unfortunately, a solution of the above stated problem is still at large. Tang, Buehrer and Lee [1982] Showed that the general problem of designing a multi-attribute file system which is optimal with respect to partial match queries is an NP-hard problem. In other words, our problem of designing an optimal multi-attribute file must be considered as a very difficult problem. Thus it is inevitable that some heuristic methods such as one proposed in [Rothnie and Lozano 1974], has to be used.

In [Lin, Lee and Du 1979], it was pointed out that all good file design methods for partial match retrieval exhibit one common property: Records in one bucket are similar to one another. It was also pointed out that multi-attribute file systems based upon multi-key hashing (MKH for short) functions proposed by [Rothnie and Lozano 1974] exhibit the property of clustering similar records together.

We shall use MKH as a means of storing and retrieving records throughout this paper.

SECTION 2. OPTIMAL MULTI-ATTRIBUTE FILE SYSTEMS BASED UPON MULTIPLE KEY HASHING FUNCTIONS

The multiple key hashing method (MKH for short) was proposed by Rothnie and Lozano [1974].

The MKH method can be briefly defined as follows:

(1) Choose a hashing function f_i for domain D_i, where D_i is the i-th attribute domain, such that $f_i: D_i \rightarrow \{1,2,\ldots,m_i\}$ where $m_1 m_2 \ldots m_N = NB$, the total number of buckets required by the file.

(2) Associate with each N-tuple $(L_1,L_2,\ldots,L_N)$ a bucket where L_i is an integer, $1 \leq L_i \leq m_i$.

(3) If the attributes $A_1,A_2,\ldots,A_N$ of a record R have values $a_1,a_2,\ldots,a_N$ respectively, assign R into the bucket associated with $(f_1(a_1),f_2(a_2),\ldots,f_N(a_N))$ where $a_i \in D_i$ for $i=1,2,\ldots,N$.

If all partial match queries are random, the retrieval of the record $(a_1,a_2,\ldots,a_N)$ (Every attribute is used.) needs examining exactly one bucket. However, the partial match query with $A_i=a_i$ (for any a_i) examines $\frac{NB}{m_i}$ buckets, the query with $A_i=a_i$ and $A_j=a_j$ (for any a_i and a_j) examines $\frac{NB}{m_i m_j}$ buckets, etc.

Let us assume that each file has N attributes and p_i, $i=1,2,\ldots,N$, is the hypothetical probability that a given attribute is specified.

We now ask: What is the number of buckets that must be examined by a partial match query of the form "$A_{i_1}=a_{i_1}$, $A_{i_2}=a_{i_2},\ldots,A_{i_j}=a_{i_j}$, $i_1 \neq i_2 \neq \ldots \neq i_j$?"

and what is the relative importance of the set of queries involve $A_{i_1}, A_{i_2}, \ldots$ and A_{i_j}?

[1] (a) There are $\frac{NB}{m_i}$ buckets that must be examined by the partial match query with $A_i = a_i$.

(b) The relative importance of the set of queries involves A_i is

$$p_i \prod_{\substack{j=1\\ j\neq i}}^{N} (1-p_j) = \frac{p_i}{1-p_i} \prod_{j=1}^{N} (1-p_j).$$

[2] (a) There are $\frac{NB}{m_i m_k}$ buckets that must be examined by the partial match query with $A_i = a_i$ and $A_k = a_k$.

(b) The relative importance of the set of queries involves A_i and A_k is

$$p_i \cdot p_k \prod_{\substack{j=1\\ j\neq i\\ j\neq k}}^{N} (1-p_j) = \frac{p_i}{1-p_i} \cdot \frac{p_k}{1-p_k} \prod_{j=1}^{N} (1-p_j).$$

[3] (a) There are $\frac{NB}{m_{i_1} m_{i_2} \ldots m_{i_{N-1}}}$ buckets that must be examined by the partial match query with $A_{i_1} = a_{i_1}$, $A_{i_2} = a_{i_2}, \ldots$ and $A_{i_{N-1}} = a_{i_{N-1}}$.

(b) The relative importance of the set of queries involves $A_{i_1}, A_{i_2}, \ldots,$ and $A_{i_{N-1}}$ is

$$p_{i_1} \cdot p_{i_2} \cdot \ldots \cdot p_{i_{N-1}} (1-p_{i_N}) = \frac{p_{i_1}}{1-p_{i_1}} \cdot \frac{p_{i_2}}{1-p_{i_2}} \cdots \frac{p_{i_{N-1}}}{1-p_{i_{N-1}}} \prod_{j=1}^{N} (1-p_j).$$

Let ENB_i, for $i=1,\ldots,N-1$, be the expected number of buckets being accessed to satisfy partial match queries with i attributes being specified.

We have the followings:

$$ENB_1 = \sum_{i=1}^{N} \left(\frac{p_i}{1-p_i} \prod_{j=1}^{N} (1-p_j)\right) \cdot \frac{NB}{m_i}$$

$$ENB_2 = \sum_{i=1}^{N-1} \sum_{k=i+1}^{N} \left(\frac{p_i}{1-p_i} \frac{p_k}{1-p_k} \prod_{j=1}^{N} (1-p_j)\right) \cdot \frac{NB}{m_i m_k}$$

$$\vdots$$

$$ENB_{N-1} = \sum_{i_1=1}^{2} \sum_{i_2=i_1+1}^{3} \cdots \sum_{i_{N-1}=N-1}^{N} \left(\frac{p_{i_1}}{1-p_{i_1}} \frac{p_{i_2}}{1-p_{i_2}} \cdots \frac{p_{i_{N-1}}}{1-p_{i_{N-1}}} \cdot \prod_{j=1}^{N} (1-p_j)\right) \cdot \frac{NB}{m_{i_1} m_{i_2} \ldots m_{i_{N-1}}}.$$

For a multi-attribute file system based upon MKH, let ANB_{MKH} be the average number of buckets to be accessed over all partial match queries. Therefore

$$ANB_{MKH} = \sum_{i=1}^{N-1} ENB_i.$$

Let $\{p_{i_1},p_{i_2},\ldots,p_{i_k}\}$ be a subset with k elements chosen from $\{p_1,p_2,\ldots,p_N\}$ and $\{m_{i_1},m_{i_2},\ldots,m_{i_k}\}$ be a subset with k elements chosen from $\{m_1,m_2,\ldots,m_N\}$. In general, there are

$$\sum_{\substack{\{p_{i_1},p_{i_2},\ldots,p_{i_k}\}\varepsilon\{p_1,p_2,\ldots,p_N\}\\ \{m_{i_1},m_{i_2},\ldots,m_{i_k}\}\varepsilon\{m_1,m_2,\ldots,m_N\}\\ i_1<i_2<\ldots<i_k}} \left(\frac{p_{i_1}}{1-p_{i_1}}\cdot\frac{p_{i_2}}{1-p_{i_2}}\cdot\ldots\cdot\frac{p_{i_k}}{1-p_{i_k}}\cdot\prod_{j=1}^{N}(1-p_j)\right)\cdot\frac{NB}{m_{i_1}m_{i_2}\ldots m_{i_k}}$$

expected number of buckets that should be examined for partial match queries which involve the i_1-th, the i_2-th,... and the i_k-th attributes.

Hence

$$ANB_{MKH} = \sum_{k=1}^{N-1}\ \sum_{\substack{\{p_{i_1},p_{i_2},\ldots,p_{i_k}\}\varepsilon\{p_1,p_2,\ldots,p_N\}\\ \{m_{i_1},m_{i_2},\ldots,m_{i_k}\}\varepsilon\{m_1,m_2,\ldots,m_N\}\\ i_1<i_2<\ldots<i_k}} \left(\frac{p_{i_1}}{1-p_{i_1}}\cdot\frac{p_{i_2}}{1-p_{i_2}}\cdot\ldots\cdot\frac{p_{i_k}}{1-p_{i_k}}\cdot\prod_{j=1}^{N}(1-p_j)\right)\cdot\frac{NB}{m_{i_1}m_{i_2}\ldots m_{i_k}} . \quad (2\text{-}1)$$

Let $P=\prod_{i=1}^{N}(1-p_i)$ and $\frac{p_i}{1-p_i}=r_i$ for $i=1,2,\ldots,N$. Then (2-1) can be rewritten as

$$ANB_{MKH} = P\cdot NB\cdot\sum_{k=1}^{N-1}\ \sum_{\substack{\{r_{i_1},r_{i_2},\ldots,r_{i_k}\}\varepsilon\{r_1,r_2,\ldots,r_N\}\\ \{m_{i_1},m_{i_2},\ldots,m_{i_k}\}\varepsilon\{m_1,m_2,\ldots,m_N\}\\ i_1<i_2<\ldots<i_k}} \frac{r_{i_1}}{m_{i_1}}\cdot\frac{r_{i_2}}{m_{i_2}}\cdot\ldots\cdot\frac{r_{i_k}}{m_{i_k}}$$

$$= P\cdot NB\cdot\left(\prod_{i=1}^{N}\left(1+\frac{r_i}{m_i}\right)-1-\prod_{i=1}^{N}\frac{r_i}{m_i}\right) . \quad (2\text{-}2)$$

By minimizing (2-2), the optimal solution of m_i is

$$m_i = \frac{\frac{p_i}{1-p_i}}{\sqrt[N]{\prod_{i=1}^{N}\left(\frac{p_i}{1-p_i}\right)\cdot\frac{1}{NB}}} = \frac{p_i}{(1-p_i)\sqrt[N]{\prod_{i=1}^{N}\left(\frac{p_i}{1-p_i}\right)\cdot\frac{1}{NB}}} .$$

The theoratically optimal performance of the multi-attribute file system based upon MKH functions now can be calculated as

$$ANB_{MKH} = \left(\prod_{i=1}^{N}(1-p_i)\right)\cdot NB\cdot\left(\prod_{j=1}^{N}\left(1+\sqrt[N]{\left(\prod_{i=1}^{N}\frac{p_i}{1-p_i}\right)\cdot\frac{1}{NB}}\right)-1-\left(\prod_{i=1}^{N}\frac{p_i}{1-p_i}\right)\cdot\frac{1}{NB}\right).$$

Unfortunately, the optimal solutions of m_i's are not always integers. Hence these solutions are not practical. In the following section, we shall discuss some theories which can guarantee optimal integer solutions under certain restricted cases.

SECTION 3. THE THEORIES OF MINIMAL N-TUPLE AND OPTIMAL NULTI-ATTRIBUTE FILES BASED UPON MKH FUNCTIONS

In this section, we shall show how the theories of minimal N-tuple [Chang, Lee and Du 1980] can be applied to design a multi-attribute file based upon MKH functions which is optimal with respect to partial match queries.

In the rest of this paper, whenever we mention an N-tuple $(a_1,a_2,\ldots,a_N)$, we shall assume that a_i is an integer. Because of space, all the proofs of lemmas and theorems in this section can not be presented. But they can be obtained by writing to the authors.

Definition [Chang, Lee and Du 1980]

An N-tuple $(a_1,a_2,\ldots,a_N)$ is called an N-tuple of C, where C is a positive integer if $\prod_{i=1}^{N} a_i=C$ and a_i is an integer.

Definition [Chang, Lee and Du 1980]

A 2-tuple (a_1,a_2) is called a minimal 2-tuple if for every other 2-tuple (a_1',a_2') where $a_1a_2=a_1'a_2'$, $a_1+a_2 < a_1'+a_2'$.

Definition [Chang, Lee and Du 1980]

An N-tuple $(a_1,a_2,\ldots,a_N)$ is called a minimal N-tuple of C, if $\prod_{i=1}^{N} a_i=C$ and for $1\leq i,\ j\leq N$, (a_i,a_j) is a minimal 2-tuple.

In [Chang, Lee and Du 1980], they showed a very important theorem as follows:

Theorem 3.1 [Chang, Lee and Du 1980]

If there is only one minimal N-tuple of an integer constant C, $\prod_{i=1}^{N}(1+z_i)$ is the smallest among all possible N-tuples of C, iff $(z_1,z_2,\ldots,z_N)$ is the only minimal N-tuple of C.

In [Chang, Lee and Du 1980], they provided a very important algorithm to find a minimal N-tuple of C.

By Theorem 3.1, the following theorem can be easily proved.

Theorem 3.2.

Let NB be the number of buckets and p_i be the hypothetical probability attached to the i-th attribute for all $i=1,2\ \ldots,N$. Let ANB_{MKH} be the average number of buckets to be examined over all partial match queries for an N-attribute file based upon MKH functions. Let $r_i=p_i/(1-p_i)$ for $1\leq i\leq N$.

The ANB_{MKH} over all possible N-tuples of NB, is minimized if the hashing function divides each D_i into m_i subsets, if $q_i=m_i/r_i$ is an integer for all $i=1,2,\ldots,N$ and $NB/\prod_{i=1}^{N} r_i$ is an integer and $(q_1,q_2,q_3,\ldots,q_N)$ is the only minimal N-tuple of $NB/\prod_{i=1}^{N} r_i$.

Let us consider the case that the hypothetical probabilities attached to the attributes are equal. That is, let $p_i=p$ for $i=1,2,\ldots,N$. Then $r_i=p_i/(1-p_i)=p/(1-p)=r$ for $i=1,2,\ldots,N$. Now the ANB_{MKH} as shown in (2-2) becomes

$$ANB_{MKH} = P\cdot NB\cdot\left(\prod_{i=1}^{N}\left(1+\frac{r}{m_i}\right) - 1 - \frac{r^N}{NB}\right)$$

$$= P\cdot NB\cdot\left(\frac{1}{NB}\prod_{i=1}^{N}(m_i + r) - 1 - \frac{r^N}{NB}\right) .$$

Therefore, $ANB_{MKH} = P\cdot \prod_{i=1}^{N}(m_i+r) - P(NB + r^N)$. (3-1)

In the following, we shall discuss the minimization of Formula (3-1) in detail.

Definition [Chang, Lee and Du 1980]

Given an N-tuple $S=(a_1,a_2,\ldots,a_N)$, $F(S,k)$, $1\le k\le N$ is defined as follows:

$$F(S,k) = \sum_{\substack{i_1 < i_2 < \ldots < i_k \\ \text{for all possible } (i_1,i_2,\ldots,i_k)\text{'s}}} a_{i_1}a_{i_2}\ldots a_{i_k}$$

Now, we define the $G(S,k,N,r)$ as follows:

Definition

Let r be a constant. Given an N-tuple $S=(a_1,a_2,\ldots,a_N)$, $G(S,k,N,r)$, $1\le k\le N$ is defined as

$$G(S,k,N,r) = \Big(\sum_{\substack{i_1 < i_2 < \ldots < i_k \\ \text{for all possible } (i_1,i_2,\ldots,i_k)\text{'s}}} a_{i_1}a_{i_2}\ldots a_{i_k}\Big)r^{N-K}$$

In other words, $G(s,k,N,r)=F(S,k)r^{N-K}$ for $k=1,2,\ldots,N$. Note that when $r=1$, $G(S,k,N,1)=F(S,k)$ for $k=1,2,\ldots,N$.

Lemma 3.1

Let $S=(a_1,a_2,\ldots,a_N)$ and $S'=(a_1',a_2',\ldots,a_N')$. Let N be an integer and r be a constant. $F(S',k) < F(S,k)$, for $k=1,2,\ldots,N-1$, iff $G(S',k,N,r) < G(S,k,N,r)$, for $k=1,2,\ldots,N-1$.

Lemma 3.2 [Chang, Lee and Du 1980]

Let $S=(a_1,a_2,\ldots,a_N)$ and $\prod_{i=1}^{N} a_i=C$. If S is not a minimal N-tuple of C, S can be transformed into $S'=(a_1',a_2',\ldots,a_N')$ such that S' is a minimal N-tuple of C and $F(S',k) < F(S,k)$ for $k=1,2,\ldots,N-1$.

Lemma 3.3

Let $S=(m_1,m_2,\ldots,m_N)$ and $\prod_{i=1}^{N} m_i=NB$. Let r be a constant and N an integer. If S is not a minimal N-tuple of NB, S can be transformed into $S'=(m'_1, m'_2,\ldots,m'_N)$ such that S' is a minimal N-tuple of NB and $G(S',k,N,r) < G(S,k,N,r)$ for $k=1,2,\ldots,N-1$.

Theorem 3.3

Let r be a constant and N an integer. If there is only one minimal N-tuple of a constant NB, $\sum_{k=1}^{N-1} G(S,k,N,r)$ is the smallest among all possible N-tuples of NB, iff S is the only minimal N-tuple of NB.

Theorem 3.4

If the probabilities attached to the attributes are equal and $(m_1,m_2,\ldots,m_N)$ is the only minimal N-tuple of NB, then $m_1,m_2,\ldots,$ and m_N are optimal solutions which minimize ANB_{MKH} over all possible N-tuples of NB.

SECTION 4. CONCLUSIONS

In this paper, we have derived the performance formula for multi-attribute files based upon MKH functions. We have provided the theoratically optimal performance of a file system based upon MKH function.

We have shown that if some restricted conditions are met, it is possible to have an optimal file for MKH.

We have also reduced the problem of designing optimal multi-attribute files based upon MKH functions, when the hypothetical probabilities attached to attributes are equal, to a problem of finding minimal N-tuples.

REFERENCES

[1] Aho, A. V. and Ullman, J. D. Optimal Partial-match Retrieval when Fields are Independently Specified, ACM Transactions on Database Systems, Vol. 4, No. 2, June 1979, pp. 168-179.

[2] Bentley, J. L. and Friedman, J. H. Data Structures for Range Searching, Computing Surveys, Vol. 11, No. 4, Dec. 1979, pp. 397-409.

[3] Bolour, A. Optimality Properties of Multiple-key Hashing Functions, Journal of the Association for Computing Machinery, Vol. 26, No. 2, April 1979, pp. 196-210.

[4] Chang, C. C. A Survey of Multi-key File Design, Proc. of The First Conference on Computer Algorithms, Hsinchu, Taiwan, July 1982, pp. 3-1-3-62.

[5] Chang, C. C. On Cartesian Product Files, Journal of Electrical Engineering, Vol. 26, No. 2, April 1983, pp. 5-23.

[6] Chang, C. C. and Lee, R. C. T. Optimal Cartesian Product Files for Partial Match Queries and Partial Match Patterns, Proc. NCS 1979 Conference, Taipei, Taiwan, Dec. 1979, pp. 5-27 - 5-37.

[7] Chang, C. C., Du, M. W. and Lee, R. C. T. Performance Analyses of Cartesian Product Files and Random Files, to appear in IEEE Transactions on Software Engineering.

[8] Chang, C. C., Du, M. W. and Lee, R. C. T. Performance Analyses of Partition Files, Proc. ICS 1982 Conference, Taichung, Taiwan, Dec. 1982, pp. 718-726.

[9] Chang, C. C., Lee, R. C. T. and Du, H. C. Some Properties of Cartesian Product Files, Proc. ACM-SIGMOD 1980 Conference, Santa Monica, Calif.,

May 1980, pp. 157-168.

[10] Chang, C. C., Lee, R. C. T. and Du, M. W. Symbolic Gray Code as a Perfect Multi-attribute Hashing Scheme for Partial Match Queries, IEEE Transactions on Software Engineering, Vol. SE-8, No. 3, May 1982, pp. 235-249.

[11] Chang, J. M. and Fu, K. S. On the Retrieval Time and the Storage Space of Doubly-chained Multiple-attribute Tree Data Base Organization, Policy Analysis and Information Systems, Vol. 1, No. 2, Jan. 1978, pp. 22-48.

[12] Ghosh, S. P. Data Base Organization for Data Management, Academic Press, N. Y., 1977.

[13] Lee, R. C. T. Clustering Analysis and its Applications, Advances in Information Systems Science, Plenum Press, N. Y., (edited by J. T. Tou) 1981, pp. 169-187.

[14] Lee, R. C. T. and Tseng, S. H. Multi-key Sorting, Policy Analysis and Information Systems, Vol. 3, No. 2, Dec. 1979, pp. 1-20.

[15] Lin, W. C., Lee, R. C. T. and Du, H. C. Common Properties of Some Multi-attribute File Systems, IEEE Transactions on Software Engineering, Vol. SE-5, No. 2, March 1979, pp. 160-174.

[16] Liou, J. H. and Yao, S. B. Multi-dimensional Clustering for Data Base Organizations, Information Systems, Vol. 2, 1977, pp. 187-198.

[17] Rivest, R. L. Partial-match Retrieval Algorithms, SIAM Journal of Computing, Vol. 15, No. 1, March 1976, pp. 19-50.

[18] Rothnie, J. B. and Lozano, T. Attribute Based File Organization in a Paged Memory Environment, Communications of the Association for Computing Machinery, Vol. 17, No. 2, Feb. 1974, pp. 63-69.

[19] Tang, T. Y., D. J. Buehrer and R. C. T. Lee, Proc. ICS 1982 Conference, Taichung, Taiwan, Dec. 1982, pp. 649-655.

NEW VARIANTS OF COALESCED HASHING*

(extended abstract)

Wen-Chin Chen
Jeffrey Scott Vitter

Department of Computer Science
Brown University
Providence, R.I. 02912

ABSTRACT

The coalesced hashing method is a very fast method for dynamic information storage and retrieval. This paper answers some open questions regarding the performance of coalesced hashing and its variants.

In all the variants, the range of the hash function is called the *address region*, and extra space reserved for storing colliders is called the *cellar*. We refer to the unmodified method, which was analyzed previously, as *late-insertion* coalesced hashing. We analyze two new variations called *early-insertion* and *varied-insertion*, and answer some open questions posed in the literature. When there is no cellar, the early-insertion method is better than late-insertion, however past experience has indicated that it might be worse when there is a cellar. Our analysis confirms that it is worse. The varied-insertion method was introduced as a means of obtaining the advantages of early-insertion and late-insertion. This paper shows that varied-insertion requires fewer probes per successful search, on the average, than does late-insertion. We conjecture that varied-insertion is optimum among all direct chaining methods, under the restrictions that the records are not moved after they are inserted, that for each chain the relative order of its records does not change after more insertions, and that only one link field is used per table slot.

Each of these three coalesced hashing methods has a parameter that relates the sizes of the address region and the cellar. Techniques are designed in this paper for tuning the parameter in order to achieve optimum search times. We conclude this paper with a list of open problems.

1. INTRODUCTION

Coalesced hashing is a very efficient technique for storing and retrieving information dynamically. This searching algorithm was introduced in [Wil59] and partly analyzed in [Knu73], [Gui76], [GK81], [Vit83], [CV83], and [Kno83]. The notation we use is that each package of information is stored in computer memory as a *record*. We assume that there is a special field in each record, called the *key*, that uniquely identifies it. Given a input K, the task of a searching algorithm is to return the record that has K as its key, if such a record exists.

Throughout this paper, the number of inserted records is denoted by N, and the number of slots in the hash table is denoted by M'. The first M slots, which serve as the range of the hash function, are called the *address region*; the remaining $M' - M$ slots make up the *cellar*. We assume that there is a pre-defined hash function

$$\textit{hash} : \{\text{all possible keys}\} \rightarrow \{1, 2, \ldots, M\} \tag{1}$$

that assigns each record to its *hash address* in a random (uniform) manner. We say that a *collision* occurs when the hash address of a record is already occupied by a previously inserted record, and the record must be inserted elsewhere. All records with the same hash address are kept chained together in a linked-list. The cellar

---

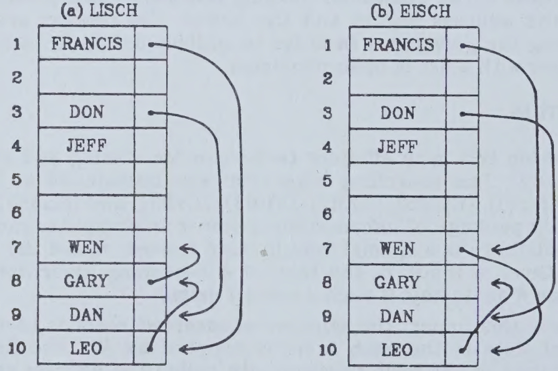
* Both authors were supported in part by an IBM research contract. The second author was also supported in part by National Science Foundation grant MCS-81-05324.

described above is devoted solely to storing the records that collide when inserted. Once the cellar gets full, subsequent colliders are stored in empty slots in the address region. The special case in which $M = M'$ and there is no cellar is called *standard coalesced hashing*. The ratios $\alpha = N/M'$ and $\beta = M/M'$ are called the *load factor* and *address factor*.

The algorithm combining storage and retrieval works as follows: Given a record with key K, the algorithm searches for it in the hash table, starting at its hash address $hash(K)$ and following the links in the chain. If the record is found, the search is successful; otherwise, the end of the chain is reached and the search is unsuccessful, in which case the record is inserted as follows: if position $hash(K)$ is empty, then the record is stored at that location; otherwise, the record is stored in the largest-numbered empty slot in the table and is linked into the chain that contains slot $hash(K)$ (at some point in the chain after slot $hash(K)$).

There are several different ways to link that record into the chain: The conventional method links the record to the *end* of chain that contains slot $hash(K)$. Another method proposed in [Vit82b] is to insert the record into the chain *immediately after* slot $hash(K)$ by rerouting pointers. This method is called *early-insertion coalesced hashing* (EICH), since the record is inserted *early* into the chain; whereas the conventional method is referred to as *late-insertion coalesced hashing* (LICH). For standard coalesced hashing (when there is no cellar), these methods are abbreviated EISCH and LISCH. Insertion can be done faster with the early-insertion method when it is known a priori that the record is not already present in the table, since it is not necessary to search to the end of the chain.

An example of these two methods for the special case of standard coalesced hashing is given in Figure 1. The record WEN collides with FRANCIS at slot 1. With the LISCH method pictured in Figure 1(a), WEN is linked to the end of the chain containing FRANCIS, whereas with EISCH in Figure 1(b), WEN is inserted into the chain at the point between FRANCIS and LEO. The average successful search time in Figure 1(b) is slightly better than in Figure 1(a), because inserting WEN immediately



Keys:		FRANCIS	DON	LEO	JEFF	DAN	GARY	WEN
Hash Addresses:	(a)(b)	1	3	1	4	3	10	1

ave. # probes per succ. search: (a) 13/7 ≈ 1.86, (b) 12/7 ≈ 1.71.
ave. # probes per unsucc. search: (a) 17/10 = 1.7, (b) 17/10 = 1.7.

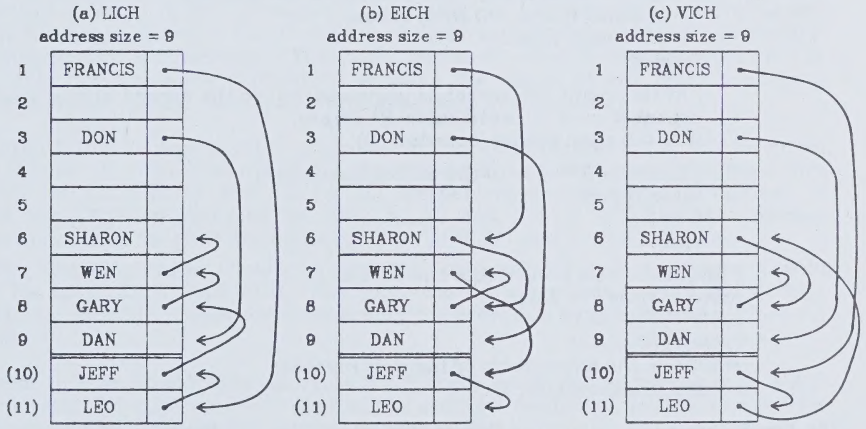
Figure 1. Standard Coalesced Hashing, M' = M = 10, N = 7. (a) LISCH, (b) EISCH.

after FRANCIS (rather than at the end of the chain) reduces the search time for WEN from four probes to two and increases the search time for LEO from two probes to three. That results in a net decrease of one probe.

Without a cellar, EISCH has faster search performance than LISCH, but when the early-insertion coalesced hashing method is used with a cellar (EICH), the average search performance is inferior to that of LICH, since in EICH the records of a chain that are in the cellar come at the end of the chain, whereas in LICH they come immediately after the first record in the chain. For example, in Figure 2(b) the insertion of WEN causes both the cellar records LEO and JEFF to move one more link further from their hash addresses. That does not happen with late-insertion in Figure 2(a).

A new variant of coalesced hashing proposed in [Vit82b], called *varied-insertion*, combines the advantages of early-insertion and late-insertion. The *varied-insertion coalesced hashing* method (VICH) is slightly different from EICH, in that the collider is inserted immediately after its hash address (as in EICH) except when the cellar is full and the hash address of the inserted record is the location of the first record in the chain. In that case, the record is inserted into the chain immediately after the last cellar slot in the chain. For the special case of standard coalesced hashing (when there is no cellar), the two methods are identical; that is, varied-insertion standard coalesced hashing (VISCH) is the same as early-insertion standard coalesced hashing (EISCH). An example of VICH is shown in Figure 2(c).

We shall see that varied-insertion is superior to both early-insertion and late-insertion with respect to search time. Based on preliminary study, we conjecture that VICH is the *search-time optimum* direct chaining method under the assumptions that the records are not moved after they are inserted, that the relative order of the records in each chain does not change after more insertions, and that each slot has only one link field.



Keys:	FRANCIS	DON	LEO	JEFF	DAN	GARY	WEN	SHARON
Hash Addresses:	1	3	1	1	3	1	8	1

ave. # probes per unsucc. search: (a) 18/9 = 2.0, (b) 24/9 ≈ 2.67, (c) 18/9 = 2.0.
ave. # probes per succ. search: (a) 21/8 ≈ 2.63, (b) 22/8 = 2.75, (c) 20/8 = 2.5.

Figure 2. Coalesced Hashing, $M' = 11$, $M = 9$, $N = 8$. (a) LICH, (b) EICH, and (c) VICH.

Theorems 1, 2, and 3 in section 3 are the main results of this paper. They give the unsuccessful and successful search times of LICH, EICH, and VICH as functions of the load factor α and the address factor β. Theorem 1 agrees with a result derived in [Vit83]; theorems 2 and 3 solve the open problems posed in [Vit82b], [CV83], and [Kno83]. These theorems are proved simultaneously in Section 4. In Section 5, we use these formulas to find the optimum address factor β_{OPT} for each of these methods, and we graph their search times assuming that β is optimized for each load factor α. The initialization $\beta = .86$ seems to be a good choice: it leads to almost optimum search performance for most load factors α. In Section 6 we discuss the conjecture about VICH search-time optimality. We make some concluding remarks and list some interesting open problems in Section 7.

2. THE FORMAL ALGORITHMS

Each algorithm searches an M'-slot hash table, looking for a given key K. By convention, if the search is unsuccessful and the table is not full, then the record with key K is inserted.

The size of the address region is M; the hash function *hash* returns a value between 1 and M (inclusive). For convenience, we make use of slot 0, which is always empty. The global variable R is used to find an empty space whenever a collision must be stored in the table. Initially the table is empty, and we have $R = M' + 1$. When an empty space is requested, R is decremented until one is found. We assume that the following initializations have been made before any searches or insertions are performed: $M := \lceil \beta M' \rceil$, for some constant $0 < \beta \leq 1$; $table[i].empty$:= **true**, for all $0 \leq i \leq M'$; and $R := M' + 1$.

```
i = hash(K);        { Compute the hash function }
if not table[i].empty then        { Search through the chain }
    begin
    repeat
        if K = table[i].key then return(successful, i);
        if table[i].link = 0 then break;
        i := table[i].link
    forever;
    { At this point the search is unsuccessful, so the record will be inserted }
    repeat R := R - 1 until table[R].empty;
    if R = 0 then return(overflow, 0);
    { Link the R^th slot to the end of the chain }
    table[i].link := R;
    i := R;
    end;
{ Insert the new record into the i^th slot }
table[i].empty := false;
table[i].key := K;
table[i].link := 0;
Initialize the other fields of the new records;
return(unsuccessful, i);
```

The **break** command causes the flow of control to leave the **repeat**-loop.

The program for EICH is an easy modification of the above code: The first line should be replaced by the double assignment

$i := j := hash(K)$;

The code to link the R^{th} record into chain should be changed from

$table[i].link := R$;

to

$table[R].link := table[j].link$;
$table[j].link := R$;

The *link* fields of all the table slots should also be initialized to 0 before any records are inserted.

The program for VICH can be obtained from the EICH program by adding the additional command

if $i > M$ **then** $j := i$

at the end of the **repeat-forever** loop.

An additional advantage of EICH and VICH over LICH is that insertions can be done very fast if it is known a priori that the record is not already in the hash table, since an unsuccessful search does not have to be performed. The code will be included in the final version of the paper.

3. THE PROBABILITY MODEL AND THE MAIN RESULTS

In this section we develop the probability model used in our analysis of EICH, LICH, and VICH, and we state our main results, Theorems 1, 2, and 3, which express the average number of probes per unsuccessful and successful search for these three methods. We will use the following parameters in our analysis:

N = the number of inserted records;
M' = the number of slots in the hash table;
M = the number of slots in the address region;
$K = M' - M$ = the number of slots in the cellar;
$\alpha = N/M'$ (load factor);
$\beta = M/M'$ (address factor).

These quantifies satisfy $0 \le N \le M'$, $0 \le \alpha \le 1$, $0 < M \le M'$, and $0 < \beta \le 1$. In the case of standard coalesced hashing (when there is no cellar), we have $M = M'$ and $\beta = 1$.

In our analysis, we will use the number of probes per search, i.e., the number of slots traversed, as a measure of search performance. Since the worst-case search times of hashing methods are very bad, we will discuss only the average-case performance here. In the average-case analysis, we assume that an unsuccessful search can begin at any of the M slots in the address region with equal probability. This includes the special case of insertion. Similarly, each of the N records in the hash table has the same probability of being the object of any given successful search. In other words, all searches and insertions are random.

Definition 1 The average number of probes in unsuccessful and successful searches in an M'-slot LICH table with address region size M and containing N records are denoted by $LC'_N(M',M)$ and $LC_N(M',M)$, respectively. The average number of probes for EICH are denoted by $EC'_N(M',M)$ and $EC_N(M',M)$, and the average number of probes for VICH are denoted by $VC'_N(M',M)$ and $VC_N(M',M)$.

The following three theorems give the unsuccessful and successful search times for LICH, EICH, and VICH, thus solve the open problems posed in [Vit82b], [CV83], and [Kno83]. These three theorems are proved in a uniform way in the next section.

Theorem 1. In an M'-slot late-insertion (LICH) hash table with address size $M = \beta M'$ that contains $N = \alpha M'$ inserted records, the average number of probes required for unsuccessful and successful searches (as $M' \to \infty$) is

$$LC'_N(M',M) \approx \begin{cases} e^{-\frac{\alpha}{\beta}} + \frac{\alpha}{\beta} & \text{if } \alpha \le \lambda\beta \\ \frac{1}{\beta} + \frac{1}{4}\left(e^{2(\frac{\alpha}{\beta} - \lambda)} - 1\right)\left(3 - \frac{2}{\beta} + 2\lambda\right) - \frac{1}{2}\left(\frac{\alpha}{\beta} - \lambda\right) & \text{if } \alpha \ge \lambda\beta \end{cases} \qquad (2)$$

$$LC_N(M',M) \approx \begin{cases} 1+\dfrac{\alpha}{2\beta} & \text{if } \alpha \le \lambda\beta \\ 1+\dfrac{\beta}{8\alpha}\left[e^{2(\frac{\alpha}{\beta}-\lambda)}-1-2(\dfrac{\alpha}{\beta}-\lambda)\right]\left(3-\dfrac{2}{\beta}+2\lambda\right) & \\ \qquad +\dfrac{1}{4}\left(\dfrac{\alpha}{\beta}+\lambda\right)+\dfrac{\lambda}{4}\left(1-\dfrac{\lambda\beta}{\alpha}\right) & \text{if } \alpha \ge \lambda\beta \end{cases} \tag{3}$$

where λ is the unique nonnegative solution to the equation

$$e^{-\lambda}+\lambda = \frac{1}{\beta}. \tag{4}$$

The error of approximation is $O(\log M'/\sqrt{M'})$ and is independent of α and β.

Theorem 2. In an M'-slot early-insertion (EICH) hash table with address size $M = \beta M'$ that contains $N = \alpha M'$ inserted records, the average number of probes required for unsuccessful and successful searches (as $M' \to \infty$) is

$$EC'_N(M',M) \approx \begin{cases} e^{-\frac{\alpha}{\beta}}+\dfrac{\alpha}{\beta} & \text{if } \alpha \le \lambda\beta \\ e^{2(\frac{\alpha}{\beta}-\lambda)}\left(\dfrac{3}{4}+\dfrac{\lambda}{2}-\dfrac{1}{2\beta}\right)+e^{(\frac{\alpha}{\beta}-\lambda)}\left(\dfrac{1}{\beta}-1\right) & \\ \qquad +\left(\dfrac{1}{4}-\dfrac{\alpha}{2\beta}+\dfrac{1}{2\beta}\right) & \text{if } \alpha \ge \lambda\beta \end{cases} \tag{5}$$

$$EC_N(M',M) \approx \begin{cases} 1+\dfrac{\alpha}{2\beta} & \text{if } \alpha \le \lambda\beta \\ 1+\dfrac{\alpha}{2\beta}+\dfrac{\beta}{\alpha}\left[(e^{(\frac{\alpha}{\beta}-\lambda)}-1)(1+\lambda)-\left(\dfrac{\alpha}{\beta}-\lambda\right)\left(1+\dfrac{\lambda}{2}+\dfrac{\alpha}{2\beta}\right)\right] & \text{if } \alpha \ge \lambda\beta \end{cases} \tag{6}$$

where λ and the error of approximation are the same as those in Theorem 1.

Theorem 3. In an M'-slot varied-insertion (VICH) hash table with address size $M = \beta M'$ that contains $N = \alpha M'$ inserted records, the average number of probes required for unsuccessful and successful searches (as $M' \to \infty$) is

$$VC'_N(M',M) = LC'_N(M',M) \tag{7}$$

$$VC_N(M',M) \approx \begin{cases} 1+\dfrac{\alpha}{2\beta} & \text{if } \alpha \le \lambda\beta \\ 1+\dfrac{\alpha}{2\beta}+\dfrac{\beta}{\alpha}\left[(e^{(\frac{\alpha}{\beta}-\lambda)}-1)(1+\lambda)-\left(\dfrac{\alpha}{\beta}-\lambda\right)\left(1+\dfrac{\lambda}{2}+\dfrac{\alpha}{2\beta}\right)\right] & \\ \qquad +\dfrac{1-\beta}{\alpha}\left[\left(\dfrac{\alpha}{\beta}-\lambda\right)-(e^{(\frac{\alpha}{\beta}-\lambda)}-1)\right] & \text{if } \alpha \ge \lambda\beta \end{cases} \tag{8}$$

Again, λ and the error of approximation are the same as those in Theorem 1.

Note that at the crossover point at which $\alpha = \lambda\beta$, both cases of each formula (2), (3), (5), (6), (7), and (8) are equal, which follows by plugging (4) into the formulas. The condition $\alpha < \lambda\beta$ indicates that (with high probability) the cellar is not yet full. In this case, the chains have not yet coalesced in the three methods, so all the formulas are identical to the search time formulas for the *separate chaining* method ([Knu73]), which is the special case of coalesced hashing for which the cellar is large enough to accommodate all the colliders. In the special case of standard coalesced hashing in which there is no cellar (i.e., $\beta = 1$ and $\lambda = 0$), the formulas agree with the results derived in [Knu73], [Vit83], [CV83], and [Kno83].

4. PROOF OF THE THEOREMS

This section gives the proofs of Theorem 1, 2, and 3, which are the main results of this paper. The theorems give the expected search performance for LICH, EICH, and VICH, respectively. In this summary version of the paper, though, the proofs are omitted for reasons of brevity.

The analysis for EICH and VICH is much harder than that for LICH, for the following reason: Every LICH table has the property that the hash address of the k^{th} non-cellar record in a chain ($k>1$) must be the location of one of the $k-1$ preceding non-cellar records in the chain. In a random LICH table, each of these $k-1$ locations can be the hash address of the k^{th} record with equal probability $1/(k-1)$. However, that is not true in a random EICH or VICH table, as Figure 3 illustrates for the special case EISCH (which is the same as VISCH) when there is no cellar. The hash address of the fourth ($k = 4$) record in a random chain of length 4 is the location of the first record in the chain with probability 2/6, it is the location of the second record with probability 1/6, and it is the location of the third record with probability 3/6. The analysis in Theorems 2 and 3 is more difficult because it must take this into account.

The analysis will be included in the complete version of the paper.

5. TUNING TO FIND OPTIMUM PERFORMANCES

The formulas derived in the previous section express the average-case search times of LICH, EICH, and VICH as functions of three variables: the load factor $\alpha = N/M'$, the address factor $\beta = M/M'$, and a new variable $\lambda = L/M$, where L is the expected number of records required to fill up the cellar. The variable λ is related to β by (4). The main goal of this section is to use these formulas to choose the optimum values of β for each load factor α that optimize the search times. The optimum choice of β will be denoted by β_{OPT}.

For lack of space in this summary version of the paper, we omit the details of how β_{OPT} is determined. The results are shown in Figure 4. We also omit the analytical and graphical comparisons of the methods, which will be included in the

Sequence of hash addresses	Order of the keys in the chain	Key that occupies the hash address of the last key in the chain	Relative position in the chain of that key
1111	adcb	a	1
1141	adbc	b	3
1144	abdc	b	2
1114	acbd	b	3
1113	acdb	a	1
1143	abcd	c	3

Figure 3. The keys $a,b,c,$ and d are inserted in that order into an EISCH (or VISCH) table containing $M' = M = 4$ slots. This table describes all six possible chains of length 4 in an EISCH table in which the keys $a,b,c,$ and d are stored in locations 1, 4, 3, and 2, respectively. The last column shows that the hash address of the last record in the chain is more likely to be the position of the third record in the chain rather than that of the second or the first.

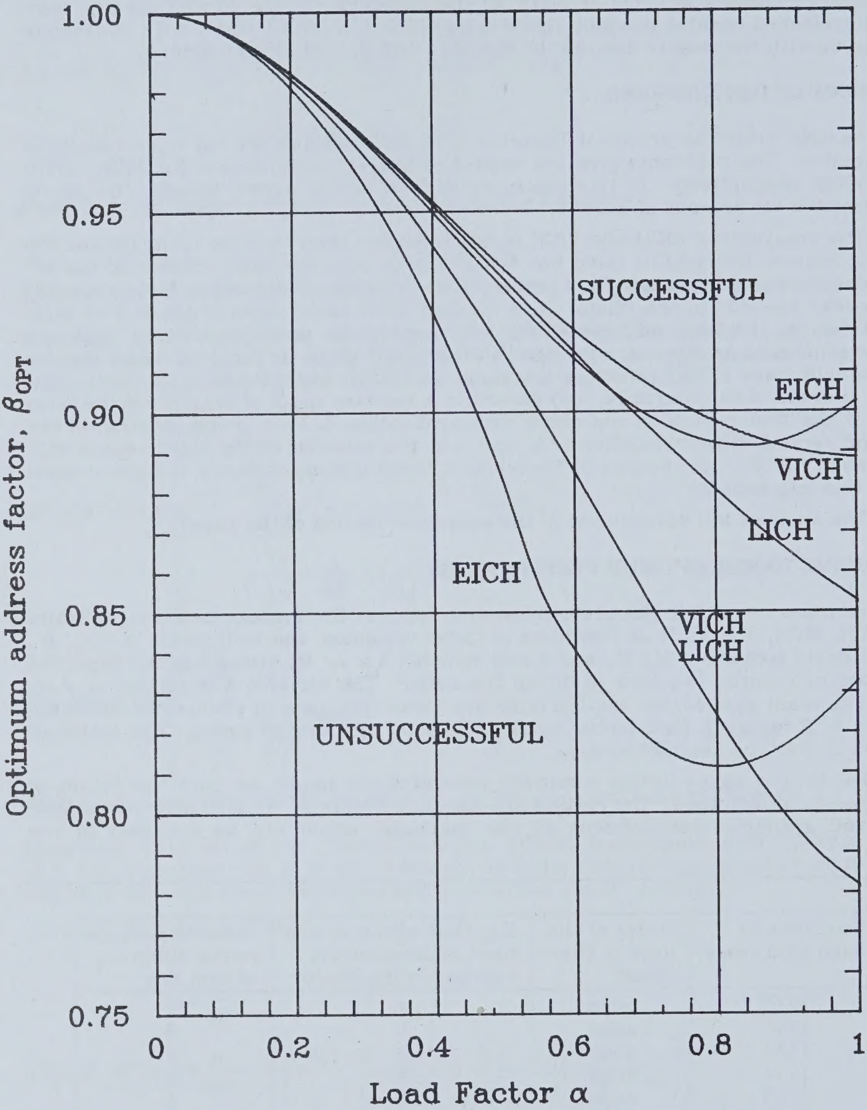


Figure 4. The β_{OPT} that optimizes search performance for LICH, EICH, and VICH.

complete version of the paper. The choice $\beta = .86$ gives near-optimum performance for a large range of load factors for each of the three variants. If some prior information about the types and frequencies of the searches is available, the choice of β can be tuned accordingly. The VICH method is the fastest of the three variants, with the LICH method a close second.

6. OPTIMALITY OF VARIED-INSERTION

In this section we investigate the search-time optimality of VICH among direct chaining methods that insert records directly into the hash table. The sizes of the address region and cellar are fixed. We assume that records cannot be moved once they are inserted. For example, the records may be "pinned" to their locations by pointers to them from outside the table. We also assume that the relative ordering of records in the chain does not change after more records are added. Finally, we assume that there is only one link field per table slot. We give evidence for the conjecture that VICH is optimum search-wise among all direct chaining methods which satisfy these conditions. The proofs are omitted in this summary version of the paper.

In coalesced hashing we always allocate available empty slots in the order M', $M' - 1$, $M' - 2$, ... For any given ordering of the available-slot list, the chains that are formed are the same regardless of which direct chaining method is used, except for the order of the individual records within the chains.

Lemma 4 shows that for successful searches, VICH is locally optimum among all direct chaining methods. VICH uses a greedy method of inserting records, in that each individual insertion is done so as to minimize the resulting average successful search time.

Lemma 4. Given a record R with key K and a chain L containing location $\mathit{hash}(K)$, a search-time optimum place to link R into L in order to minimize the average successful search times on L', where L' is the chain obtained by linking R into L, is immediately before the first non-cellar slot following slot $\mathit{hash}(K)$ in the chain, or else at the end of the chain if no such non-cellar slot exists.

The following lemma shows that in the case in which there is no cellar, then VISCH is not only locally optimum, but also globally optimum.

Lemma 5. Assume that there is no cellar. For any set of records R_1, R_2, ..., R_k that forms a chain, with the assumption that all possible hash sequences h_1, h_2, ..., h_k are equally likely, the average unsuccessful and successful search times on the chain are optimized by the VISCH (= EISCH) ordering of the records in the chain.

The next lemma generalizes the results of Lemma 5. It shows that when the cellar slots get priority on the available-slot list, VICH is globally optimum.

Lemma 6. Assume that the cellar slots are given priority on the available-slot list. For any set of records R_1, R_2, ..., R_k that forms a chain, with the assumption that all possible hash addresses h_1, h_2, ..., h_k are equally likely, the average unsuccessful and successful search times on the chain are optimized by the VICH ordering of the records in the chain.

The only thing that remains is to show that the best way to order the available-slot list is for all the cellar slots to have higher priority than the address region slots. This seems intuitively true, but so far has not been proved. If it is proved, then by lemmas 6, we can prove the following conjecture that VICH is globally optimum among all pinned direct chaining methods.

Conjecture. Varied-Insertion Coalesced Hashing gives the optimum expected search times among all chaining methods, with the probability model that all hash sequences are equally likely, under the assumptions that the records are inserted directly into the table and are not moved once they are inserted, that for each chain the relative ordering of the records does not change after more insertions, and that there in only one link field per table slot.

7. CONCLUSIONS AND OPEN PROBLEMS

We have analyzed three variants of coalesced hashing, namely, late-insertion (LICH), early-insertion (EICH), and varied-insertion (VICH), and have derived the average unsuccessful and successful search times as a function of the load factor α and the address factor β. Techniques have been developed to choose the cellar size that yields the optimum search performance. The VICH method results in the fewest probes per search, on the average, among the three variants. We conjecture that VICH is optimum among all direct chaining methods, under the assumptions that the records are not moved once they are inserted, that for each chain the relative order of its records does not change after further insertions, and that there is only one link field per table slot.

There are still some interesting open problems concerning coalesced hashing. One of them is to find a deletion algorithm that *preserves randomness*. Preserving randomness is a very desirable property of deletion algorithms. It means that deleting a record is in some sense like never having inserted it. In particular, the formulas for the average search times after N random insertions intermixed with d deletions are the same as the formulas for the average search times after $N-d$ random insertions. The formal notion of what it means to preserve randomness is defined in [Vit82a].

A deletion algorithm for coalesced hashing is given in [Vit82a] and shown to preserve randomness for late-insertion standard coalesced hashing (LISCH). There is currently no known deletion algorithm that preserves randomness for the VISCH, LICH, EICH, and VICH methods. An interesting open question is whether such algorithms exist. Unfortunately, it seems that in order to preserve randomness, a deletion algorithm must relocate some records from time to time, which may not be possible if the records are "pinned" to their locations and are not allowed to be moved. Deletion algorithms that do not relocate records (and do not preserve randomness) should therefore also be studied. One open problem is to determine how the average search times are affected by deletion algorithms that do not preserve randomness.

Another challenging open problem is to investigate coalesced hashing for the case of external searching, in which the hash table does not fit entirely in internal memory and must be partitioned into pages. Some methods have been proposed in [Vit82b] for dealing with this problem.

REFERENCES

[CV83] W. C. Chen and J. S. Vitter. Analysis of Early-Insertion Standard Coalesced Hashing. *SIAM Journal on Computing*, 12(4), November 1983.

[GK81] D. H. Greene and D. E. Knuth. *Mathematics for the Analysis of Algorithms*. Birkhauser, Boston, 1981.

[Gui76] L. J. Guibas. *The Analysis of Hashing Algorithms*. PhD dissertation, Computer Science Department, Stanford University, Technical Report STAN-CS-76-556, August 1976.

[Kno83] G. D. Knott. Direct Chaining with Coalesced Lists. To appear in *Journal of Algorithms*.

[Knu73] D. E. Knuth. *The Art of Computer Programming*. Volume 3: *Sorting and Searching*. Addison-Wesley, Reading, Massachusetts, 1973.

[Vit82a] J. S. Vitter. Deletion Algorithms for Hashing that Preserve Randomness. *Journal of Algorithms*, 3, September 1982, pp. 261-275.

[Vit82b] J. S. Vitter. Implementations for Coalesced Hashing. *Communications of the ACM*, 25(12), December 1982, pp. 911-926.

[Vit83] J. S. Vitter. Analysis of the Search Performance of Coalesced Hashing. *Journal of the ACM*, 30(2), April 1983, pp. 231-258.

[Wil59] F. A. Williams. Handling Identifiers as Internal Symbols in Language Processors, *Communications of the ACM*, 2(6), June 1959, pp. 21-24.

AN EXPLICIT SELF-TUNING ALGORITHM BASED ON NONPARAMETRIC IDENTIFICATION

R.BARS
Department of Automation, Technical University of Budapest, Hungary

L.KEVICZKY
Process Control Department, Computer and Automation Institute of the Hungarian Academy of Sciences, Budapest, Hungary

K.S.P.KUMAR
Department of Electrical Engineering, University of Minnesota,
Minneapolis, MN, USA

ABSTRACT

Control algorithms based on zero cancellation of the discrete transfer function of a sampled continuous process result in an unstable behavior in the case when the discrete transfer function has zeros outside of the unit circle. Cancellation of zeros in some regions inside of the unit circle can cause intersampling oscillations.

Several suboptimal methods were suggested to avoid this problem.

As a contribution to the subject a selftuning algorithm is developed here based on nonparametric description of the system. In this case an appropriate truncation of the model offers a suboptimal algorithm avoiding the inverse unstable behaviour.

INTRODUCTION

The application of microcomputers in control technology gives a special importance to the design of discrete-data control systems. Besides the discretized PID algorithms good behavior can be obtained using the idea of dead-beat control. Another possibility is provided by self-tuning controls. Here a model-reference predictive self-tuning control algorithm is presented based on nonparametric description of the system.

MODEL-REFERENCE PREDICTIVE CONTROL

An optimal control strategy can be given equating the predicted system output $y(t+d|t)$ and the desired reference model output $y_{rm}(t+d)$ starting from the time point t+d. Here t denotes the discrete time and d is the discrete delay. The system output is supposed to be predicted at the point t+d from the information available on the input and output signals up to the point t.

$$y(t+d|t) = y_{rm}(t+d) \tag{1}$$

The reference model output is obtained by filtering a y_r reference signal:

$$y_{rm}(t) = \frac{N(z^{-1})}{D(z^{-1})} z^{-d} y_r(t) \tag{2}$$

where N and D denote the numerator and the denominator of the filter and z^{-1} is the backward shift operator.

SYSTEM DESCRIPTION

The system is described by its discrete weighting function $W(z^{-1})$, which is the sampled response of the system for a unit pulse input of the width h, with h being the sampling interval (Fig.1).

The output signal can be expressed by the convolution sum as follows:

$$y(t) = W(z^{-1}) z^{-d} u(t) \tag{3}$$

where

$$W(z^{-1}) = w_o + w_1 z^{-1} + w_2 z^{-2} + \dots \tag{4}$$

So (3) can be written as

$$y(t+d|t) = w_o u(t) + w_1 u(t-1) + w_2 u(t-2) + \dots \tag{5}$$

This system description is nonparametric, as an infinite number of the parameters is required to determine the accurate output value. It is predictive, as the information on the input signal up to point t is sufficient to predict the output value at the point t+d.

(The system generally is given by its discrete transfer function, and its output signal is expressed as

$$y(t) = \frac{B(z^{-1})}{A(z^{-1})} z^{-d} u(t) = \frac{b_o + b_1 z^{-1} + \ldots + b_{n-1} z^{-(n-1)}}{1 + a_1 z^{-1} + \ldots + a_n z^{-n}} z^{-d} u(t) \tag{6}$$

This system description is parametric, but nonpredictive. It has to be mentioned, that with an appropriate transformation it can be brought to a predictive form.)

In recent years J.Richalet and his coworkers developed a heuristic algorithmic control based on nonparametric system description. The realizations in several industrial environments considered in [1, 2, 3] have shown a high degree of stability and robustness. These successful results encouraged the investigation of model-reference predictive control algorithms using the weighting function description of the system.

THE CONTROL ALGORITHM

Substituting (2) and (5) into (1) we have

$$w_o u(t) + w_1 u(t-1) + w_2 u(t-2) + \ldots = \frac{N}{D} y_r(t) \tag{7}$$

From here the control signal can be obtained as

$$u(t) = \frac{1}{w_o} \left[\frac{N}{D} y_r(t) - w_1 u(t-1) - w_2 u(t-2) - \ldots \right] \tag{8}$$

The control scheme is shown in Fig.2.

Its closed-loop equivalent effective in noisy cases too can also be given (Fig.3). This control algorithm ensures a stable behavior for the control system if the roots of the characteristic equation

$$w_o + w_1 z^{-1} + w_2 z^{-2} + \ldots = 0 \tag{9}$$

are inside of the unit circle.

(With the parametric system description the control signal can be expressed as:

$$u(t) = \frac{N}{D} \cdot \frac{A}{B} y_r(t) \tag{10}$$

This means, that the numerator of the transfer function is cancelled. If equation

$$B = 0 \tag{11}$$

has roots outside of the unit circle, increasing oscillations will occur between the sampling points. This problem of inverse unstability can be avoided by several suboptimal solutions [4, 5, 6].) If equation (11) has "bad" zeros, equation (9) has bad zeros too. So the nonparametric system description in itself does not give a solution to this stability problem.

The phenomenon of the inverse unstability in the nonparametric case is hoped to be avoided by an appropriate truncation of the discrete weighting function.

EFFECTS OF THE TRUNCATION OF THE WEIGHTING FUNCTION

In practical cases in control algorithm (8) only a finite number of the weighting function points is taken into account. It is worthwile to investigate the effect of the truncation to the location of the roots of the finite versions of equation (9):

$$\begin{aligned} r_1(z) &= w_o z + w_1 = 0 \\ r_2(z) &= w_o z^2 + w_1 z + w_2 = 0 \\ &\vdots \\ r_{i+1}(z) &= r_i(z).z + w_{i+1} = 0 \end{aligned} \tag{12}$$

The location of the roots depends highly on the relative values w_i/w_o. The roots were calculated for different first, second and third order examples. Fig.4 shows as an example the absolute value of the roots of a second order system versus N, where N is the degree of equations (12). With more and more weighting function points the roots tend into the inside of the unit circle. But this trend is quite weak. In the case of third order systems even a large number of the weighting function points can not ensure a stable behavior (in the parametric description in this case B also contains a zero outside of the unit circle, so with algorithm (10) the control system is also unstable). As generally $\lim_{i\to\infty} w_{i+1}=0$, the last points hardly influence the location of the roots.

The effect of the truncation of the weighting function at its beginning.

If the value of the first weighting function point w_o is low, the root of equation $r_1(z)=0$ will be outside of the unit circle. Neglecting the first point and introducing by this an extra delay promises a considerable improvement in the root location. This distortion of the weighting function is shown in Fig.5. Of course the last points of the weighting function are also truncated.

Now the model-reference design criterion is

$$y(t+d+1|t) = y_{rm}(t+d+1) \tag{13}$$

The output at the point t+d+1 can be predicted now as

$$y(t+d+1|t) = w_1\, u(t) + w_2\, u(t-1) + w_3\, u(t-2) + \dots \tag{14}$$

The control algorithm is given by

$$u(t) = \frac{1}{w_1}\left[\frac{N}{D}\, y_r(t) - w_2\, u(t-1) - w_3\, u(t-2) - \dots\right] \tag{15}$$

The control scheme is shown in Fig.6.

A static correction factor c is necessary to compensate the change in the gain caused by neglecting the first point.

$$c = 1 - \frac{w_o}{\sum_{j=0}^{\infty} w_j} \tag{16}$$

A closed-loop equivalent of the scheme can also be given similarly to Fig.3.

The stability of the algorithm now depends on the roots of equation

$$w_1 + w_2 z^{-1} + w_3 z^{-2} + \dots = 0 \tag{17}$$

Fig.7 as an example shows the absolute value of the roots of equation (17) versus N in the case of a third order system, which is inverse unstable with the original algorithm. It is seen, that for a sampling period of 5 sec with more than 5 weighting function points the modified algorithm gives a stable behavior. In further investigations 10 points of the weighting function were taken into account. Fig.8 shows the output and the control signals of this third order system for a unit step reference input.

Fig.9 illustrates the noise rejection of the closed-loop version of the control system for a unit step output noise.

The effectiveness of this simple modification in the weighting function can be explained by giving the parametric equivalent of the scheme shown is Fig.6 taking into account the relationship between the w_i weighting function points and the coefficients of the polynomials B and A:

$$\begin{aligned} w_o &= b_o \\ &\vdots \\ w_k &= b_k - \sum_{i=1}^{k} a_i w_{k-i} \end{aligned}$$

$$\begin{aligned} &\text{if} \quad k > n-1 \quad b_k=0 \\ &\text{if} \quad i > n \quad a_i=0 \end{aligned} \tag{18}$$

The equivalent parametric structure is given in Fig.10.

It is seen, that this algorithm does not compensate B, but lets it appear in the resulting function, and introduces poles, which are the roots of equation

$$\hat{B} = B - b_o A = 0 \tag{19}$$

The roots of this equation determine the stability of the control system. Further investigations proved, that in the case of second and third order systems the location of the roots of $\hat{B}$ are much better than that of B.

REFERENCES

[1] J. Richalet, A. Rault, J. L. Testud, J. Papon: Algorithmic Control of Industrial Processes, 4th IFAC Symposium on Identification and System Parameter Estimation, Tbilisi (USSR), 1976.

[2] J. Richalet, A. Rault, J. L. Testud, J. Papon: Model Predictive Heuristic Control: Applications to Industrial Processes, Automatica, Vol. 14, pp. 413-428.

[3] R. Rouhani, R. K. Mehra: Model Algorithmic Control (MAC), Basic Theoretical Properties, Automatica, Vol. 18, No. 4, pp. 401-414, 1982.

[4] Isermann, R.: Digital Control Systems, Springer Verlag, Berlin, New York, 1980.

[5] Tuschák, R.: Practical Aspects of the Design of Sampled Data Control Systems, Report of University of Minnesota, Dept. of Electrical Engineering, May, 1980.

[6] Hetthéssy, J., Keviczky, L.: Adaptive Regulators for the Deterministic Servo-regulatory Problem, Report of University of Minnesota, Dept. of Electrical Engineering, February, 1983.

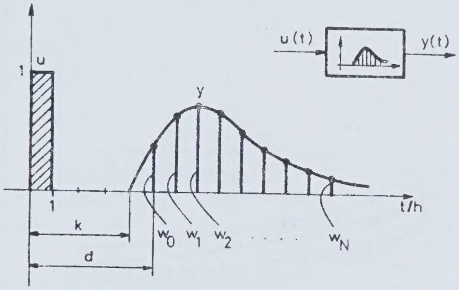


Fig.1. The discrete weighting function

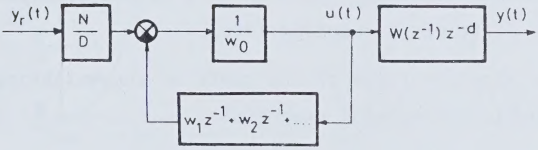

$y_r(t)$
$\frac{N}{D}$
$\frac{1}{w_0}$
u(t)
$W(z^{-1})z^{-d}$
y(t)
$w_1z^{-1}+w_2z^{-1}+\dots$

Fig.2. Open-loop control scheme

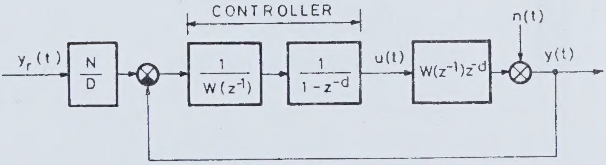

CONTROLLER
n(t)
$y_r(t)$
$\frac{N}{D}$
$\frac{1}{W(z^{-1})}$
$\frac{1}{1-z^{-d}}$
u(t)
$W(z^{-1})z^{-d}$
y(t)

Fig.3. The equivalent closed-loop control scheme

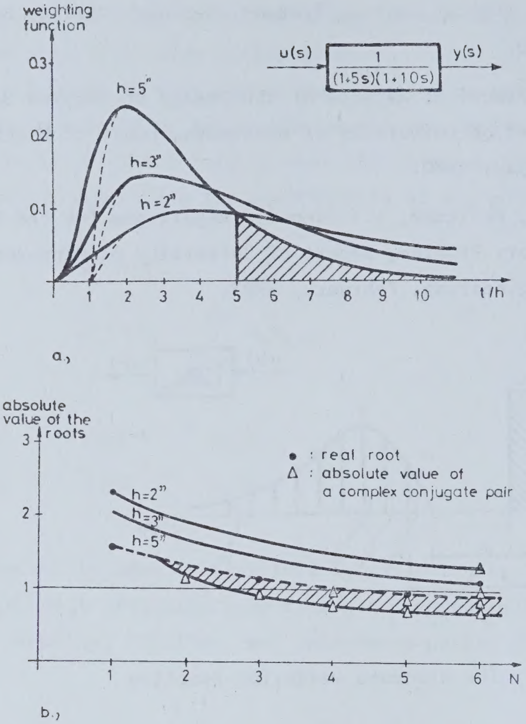


Fig.4. a) Discrete weighting function of a second order element for different h sampling times

b) The absolute value of the roots of the weighting function polynomial equation

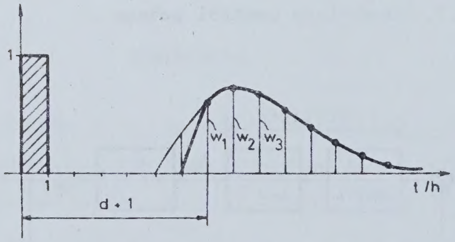


Fig.5. The modified weighting function

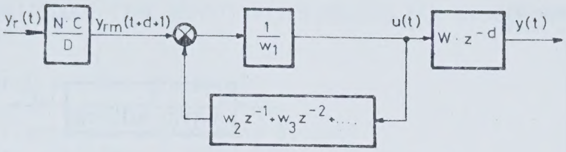

$y_r(t)$
$\frac{N \cdot C}{D}$
$y_{rm}(t+d+1)$
$\frac{1}{w_1}$
$u(t)$
$W \cdot z^{-d}$
$y(t)$
$w_2 z^{-1} + w_3 z^{-2} + \dots$

Fig.6. The modified control scheme

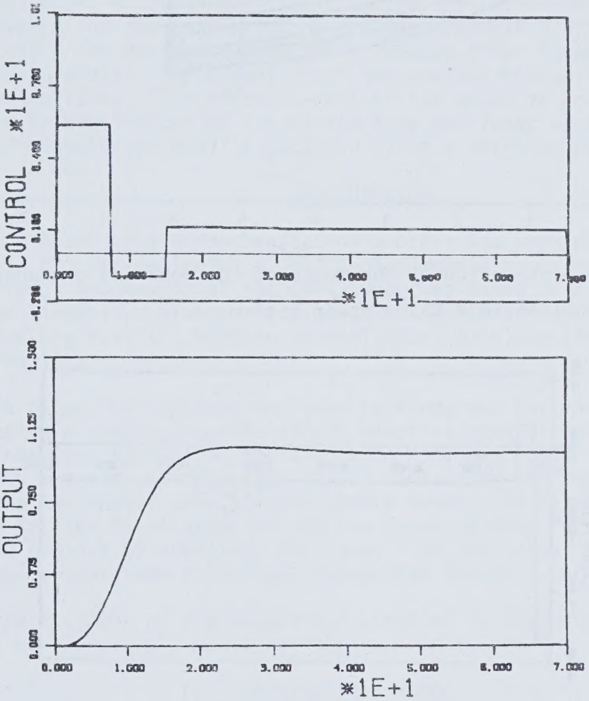


Fig.9. Output and control signals of a third order system for a unit-step output noise

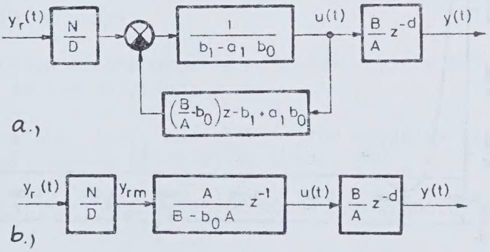

$y_r(t)$
$\frac{N}{D}$
$\frac{1}{b_1 - a_1 b_0}$
$u(t)$
$\frac{B}{A} z^{-d}$
$y(t)$
$\left(\frac{B}{A} - b_0\right) z - b_1 + a_1 b_0$
a.)
$y_r(t)$
$\frac{N}{D}$
y_{rm}
$\frac{A}{B - b_0 A} z^{-1}$
$u(t)$
$\frac{B}{A} z^{-d}$
$y(t)$
b.)

Fig.10. The equivalent parametric forms of the modified control algorithm

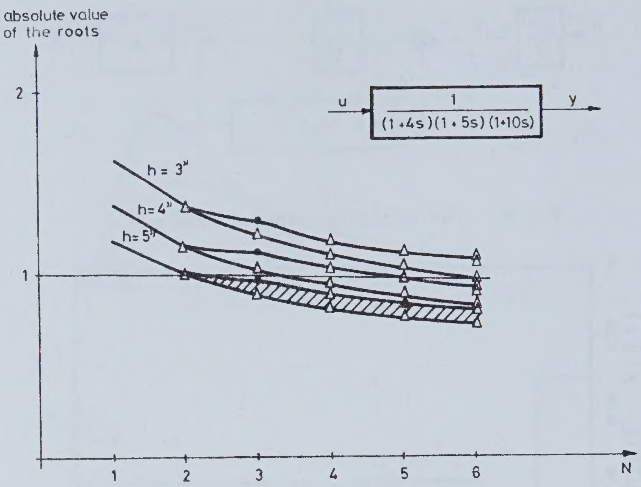


Fig.7. Absolute value of the roots of the modified characteristic equation in a third order system

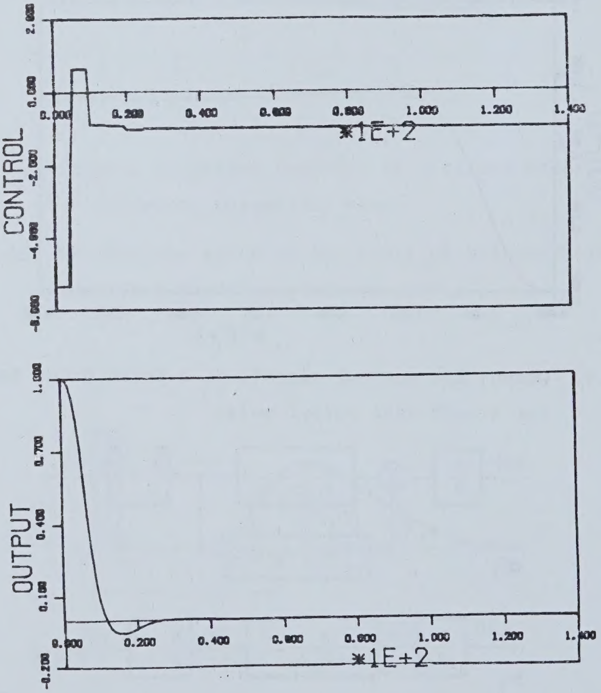


Fig.8. Output and control signals of a third order system for a unit-step reference input

DESIGN OF A CONTINUOUS-TIME ADAPTIVE CONTROLLER FOR AN ANAEROBIC FERMENTATION PROCESS*

P.A. IOANNOU and PH. OPDENACKER
University of Southern California
Dept. of Electrical Engineering-Systems
Los Angeles, California 90089-0781

ABSTRACT

In this paper, a continuous-time adaptive controller of the direct-type is designed for the regulation of the gas production of a biomethanization process. The design is based on a reduced-order linear model of the nonlinear process. The stability of the overall closed-loop linearized system is established. The effectiveness of the adaptive controller is demonstrated by simulations of the closed-loop nonlinear system and is compared with simulation results obtained using a self-tuning regulator [1].

I. INTRODUCTION

In [2], a model of a biomethanization process was obtained which was later used in [1] for designing a self-tuning regulator to regulate the gas production. The design of the self-tuner was based on a reduced order model of the linearized plant. In simulations [1] this simplified design gave satisfactory results, however no analytical justification was given for these results.

In this paper, we consider the same problem, but instead of a self-tuner we design a continuous-time direct adaptive controller. We first obtain a linearized system for the 5th order nonlinear model of the biomethanization process which we simplify by neglecting the two fast stable modes and the two weakly unobservable stable modes. An adaptive controller is designed for the first order system and Lyapunov-type functions are used to establish regions of stability and bounds for the output and parameter errors of the closed-loop full-order linearized system.

The effectiveness of the adaptive controller is demonstrated by simulations of the overall nonlinear closed-loop system.

II. MATHEMATICAL MODEL

The biomethanization process can be illustrated by the following flowchart. The influent made of waste water (volatile solids) is transformed into depolluted water and gaseous effluent, i.e., methane (CH_4) and carbon dioxide (CO_2) in three steps:

(A) Liquefaction of the organic materials by the acidogenic bacteria present in the solution,

(B) Acidification, i.e., the solubilized organics are fractioned into small chains of volatile acids, and

(C) Methanization: The volatile acids are transformed into CH_4 and CO_2.

---

*This work was supported by the U.S.C. Faculty Research and Innovation Fund, University of Southern California, Los Angeles, CA 90089.

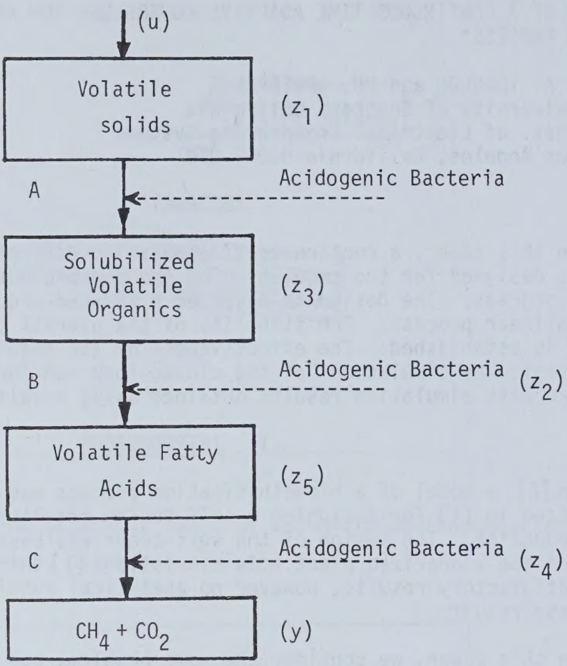

(u)
Volatile solids
(z_1)
A
Acidogenic Bacteria
Solubilized Volatile Organics
(z_3)
B
Acidogenic Bacteria
(z_2)
Volatile Fatty Acids
(z_5)
C
Acidogenic Bacteria
(z_4)
$CH_4 + CO_2$
(y)

Fig. 1. Flowchart of the biomethanization process.

The process A,B,C takes place in a digester which is assumed to be continuously fed, completely mixed and at constant temperature.

From the systems point of view, we will consider as output of the process the daily production of CH_4/liter of digester, say, y. As an input to the system, we consider the concentration of the influent in volatile solids, say, u.

A nonlinear model describing the process is given in [1] and is summarized below:

$$\dot{z}_1 = -bz_1 - z_1 z_2 + \alpha bu \tag{2.1}$$

$$\dot{z}_2 = \left(\frac{\mu_1 z_3}{1+z_3} - k_o - b\right) z_2 \tag{2.2}$$

$$\dot{z}_3 = -bz_3 + z_1 z_2 - \frac{\mu_1}{k_1} \frac{z_2 z_3}{(1+z_3)} \tag{2.3}$$

$$\dot{z}_4 = \left(\frac{\mu_2 z_5}{1+z_5} - k_2 - b\right) z_4 \tag{2.4}$$

$$\dot{z}_5 = -bz_5 + k_3 \mu_1 \frac{z_3 z_2}{1+z_3} \frac{1}{k_4} \frac{\mu_2 z_5 z_4}{1+z_5} \tag{2.5}$$

$$y = \frac{\mu_2 z_5 z_4}{1+z_5} \tag{2.6}$$

where

z_1 is the concentration of volatile solids in the fermenter

z_2 is the concentration of acidogenic bacteria in the fermenter

z_3 is the concentration of solubilized volatile organics in the fermenter

z_4 is the concentration of methanigenic bacteria in the fermenter

z_5 is the concentration of volatile fatty acids in the fermenter

b is the dilution rate and μ_1, μ_2, k_o to k_4 are constants.

In [3] a crude identification procedure based on experimental data led to the following estimate of the model parameters: $k_o = 0.02$, $k_1 = 0.006$, $k_2 = 0.02$, $k_3 = 1.1$, $k_4 = 40.$, $\mu_1 = \mu_2 = 0.4$. Adopting the values of these parameters, the following steady-state can be computed around nominal values for b and u, 0.05 (day) and 50 (g/l) respectively: $\bar{z}_1 = 11.695$, $\bar{z}_2 = 0.375$, $\bar{z}_3 = 0.212$, $\bar{z}_4 = 16.445$, $\bar{z}_5 = 0.212$. Linearizing the system of non-linear equations around this equilibrium yields the following transfer function

$$\frac{y(s)}{\mu(s)} = \frac{0.6842\,(s+0.07)^2}{(s+0.07)(s+0.075)(s+4.04984)(s+17.1786)(s+0.35055)} \tag{2.7}$$

Clearly, we can reduce the order of the system by cancelling one pole and one zero at 0.07.

Let ε and μ denote the difference between the pole at 0.075 and the remaining zero at 0.07, and the ratio $\frac{0.35055}{4.04984}$, respectively. Then we can put the model in the usual singular perturbation form as follows:

$$\dot{x} = -a_1 x + \varepsilon z + A_{12}\eta + b_o u \tag{2.8}$$

$$\dot{z} = -a_2 z - A_{12}\eta - b_o u \tag{2.9}$$

$$\mu\dot{\eta} = A_{22}\eta + \mu B_1 \dot{u} \tag{2.10}$$

$$y = x \tag{2.11}$$

where $\eta \in R^2$, $\mathrm{Re}\lambda(A_{22}) < 0$, $a_1 > 0$, $a_2 > 0$ and $b_o > 0$.

Since μ and ε are small, the system (2.8) to (2.11) can be approximated by the first order model

$$\dot{x}_o = -a_1 x_o + b_o u \tag{2.12}$$

$$y = x_o \tag{2.13}$$

In the following section, an adaptive controller designed for the simplified model (2.12), (2.13) is applied to the higher order system (2.8) to (2.11) and bounds for stability are obtained.

III. ADAPTIVE CONTROL DESIGN AND STABILITY ANALYSIS

The need for adaptive control arises from the fact that the parameters of the nonlinear system and therefore the nominal state (2.7) around which the linearized model (2.8) is obtained can change with time. The parameters of the controller will be continuously adjusted so that despite the parameter fluctuations, the nonlinearities and other uncertainties the output of the plant (daily gas production) y tracks the output y_m of a given first-order reference model

$$\dot{y}_m = -a_m y_m + b_m r , \quad a_m, b_m > 0 \tag{3.1}$$

where r is a reference input signal. To achieve this objective the following adaptive controller is designed [3]:

$$\text{Controller:} \quad u = -k_1 y - k_2 r \tag{3.2}$$

$$\text{Adaptive laws:} \quad \dot{k}_1 = -\sigma_1 k_1 + \gamma_1 ey, \; e \overset{\Delta}{=} y - y_m \tag{3.3}$$

$$\dot{k}_2 = -\sigma_2 k_2 + \gamma_2 er \tag{3.4}$$

where $\sigma_1 > 0$, $\sigma_2 > 0$, and $\gamma_1 > 0$, $\gamma_2 > 0$ are the adaptive gains.

Remark: It can be shown [3] that for $\sigma_1 = \sigma_2 = 0$ the adaptive controller (3.2) to (3.4) applied to the first order system (2.13), (2.14) guarantees that $k_1(t)$, $k_2(t)$, $x_o(t)$ are bounded and $\lim_{t\to\infty} |x_o(t) - y_m(t)| = 0$. Such a result, however, cannot be attained if the reduced controller (3.2) to (3.4) with $\sigma_1 = \sigma_2 = 0$ is applied to the full-order system (2.8) to (2.11). In [4]-[6], bounds for stability are obtained in the presence of unmodeled fast [4,5] and weakly observable [6] stable dynamics separately, by choosing $\sigma_1, \sigma_2 > 0$. In our case, both fast and weakly observable dynamics are present and σ_1, σ_2 are chosen to be positive design parameters.

The following differential equations describe the stability properties of controller (3.2) to (3.4) when applied to the linearized model (2.9) to (2.12)

$$\dot{e} = -a_m e - b_o(k_1 - k_1^*)y - b_o(k_2 - k_2^*)r + \varepsilon z + A_{12}\eta \tag{3.5}$$

$$\dot{z} = -a_2 z - A_{12}\eta + b_o k_1 (e + y_m) + b_o k_2 r \tag{3.6}$$

$$\mu\dot{\eta} = A_{22}\eta + \mu B_1 \dot{u} \tag{3.7}$$

$$\dot{k}_1 = -\sigma_1 k_1 + \gamma_1 ey \tag{3.8}$$

$$\dot{k}_2 = -\sigma_2 k_2 + \gamma_2 er \tag{3.9}$$

where $k_1^* = \dfrac{a_m - a_1}{b_o}$ and $k_2^* = -\dfrac{b_m}{b_o}$.

For convenience of analysis [7] we define

$$\xi = e - \mu A_{12} A_{22}^{-1} \eta \tag{3.10}$$

and transform (3.5) to (3.9) into

$$\dot{\xi} = -a_m \xi - b_o(k_1 - k_1^*)y - b_o(k_2 - k_2^*)r + \varepsilon z + \mu f_1 \tag{3.11}$$

$$\dot{z} = -a_2 z + b_o k_1(\xi + y_m) + b_o k_2 r + f_2 \tag{3.12}$$

$$\mu\dot{\eta} = A_{22}\eta + \mu B_2 \dot{u} \tag{3.13}$$

$$\dot{k}_1 = -\sigma_1 k_1 + \gamma_1(\xi + \mu A_{12} A_{22}^{-1} \eta)y \tag{3.14}$$

$$\dot{k}_2 = -\sigma_2 k_2 + \gamma_2(\xi + \mu A_{12} A_{22}^{-1} \eta)r \tag{3.15}$$

where $f_1 = -A_{12} A_{22}^{-1}(a_m \eta + B_1 u)$ and $f_2 = -(A_{12} - \mu b_o k_1 A_{12} A_{22}^{-1})\eta$.

Theorem: There exists a $\rho^* > 0$ such that for all $\rho \in [0, \rho^*]$, where $\rho = \max[\varepsilon, \mu]$, the solution $\xi(t)$, $z(t)$, $\eta(t)$, $k_2(t)$ of (3.11) to (3.15) starting from

$$D(\rho) = \{\xi, z, n, k_1, k_2 : |\xi| + |k_2| + |k_1| < q_1 \rho^{-\alpha},$$

$$\|\eta\| < q_2 \mu^{-\frac{1}{2}} \rho^{-\alpha},\ |z| < q_3 \varepsilon^{-\frac{1}{2}} \rho^{-\alpha}\} \tag{3.16}$$

where q_1 to q_3 are some finite positive constants, is bounded and

$$\lim_{t\to\infty} \sup_t [\xi^2(t) + \frac{b_o}{\gamma_1}(k_1(t) - k_1^*)^2 + \frac{b_o}{\gamma_2}(k_2(t) - k_2^*)^2 + \mu\eta(t)^T P \eta(t)$$

$$+ \varepsilon z(t)^2] < \frac{2}{\sigma}(\frac{b_o}{\gamma_1} \sigma_1 k_1^{*2} + \frac{b_o}{\gamma_2} \sigma_2 k_2^{*2} + \mu^2 d_{11}) \tag{3.17}$$

Furthermore, $\sigma_1, \sigma_2 > O(\rho)$ and $\sigma = \min[a_m, \sigma_1, \sigma_2]$.

Proof: Choose the positive definite function

$$V = \frac{\xi^2}{2} + \frac{b_o}{\gamma_1}\frac{(k_1 - k_1^*)^2}{2} + \frac{b_o}{\gamma_2}\frac{(k_2 - k_2^*)^2}{2} + \mu\eta^T \frac{P}{2}\eta + \frac{\varepsilon}{2} z^2 \tag{3.18}$$

where $P = P^T > 0$ and $A_{22}^T P + P A_{22} = -Q$, $Q = Q^T > 0$. Along the solution of (3.11) to (3.15) we have

$$\dot{V} = -a_m \xi^2 - \varepsilon a_2 z^2 - \frac{b_o}{\gamma_1}\sigma_1 k_1(k_1 - k_1^*) - \frac{b_o}{\gamma_2}\sigma_2 k_2(k_2 - k_2^*)$$

$$- \frac{1}{2}\eta^T Q \eta + \varepsilon z \xi + \mu[A_{12} A_{22}^{-1} B_1 \xi \dot{u} - \eta^T P B_1 \dot{u}$$

$$+ b_o(k_1 - k_1^*) A_{12} A_{22}^{-1} \eta_y + b_o(k_2 - k_2^*) A_{12} A_{22}^{-1} \eta r] \tag{3.19}$$

Now let us consider the closed surface $S(\rho, \alpha, c_o)$ defined by

$$V = c_o \rho^{-2\alpha} \tag{3.20}$$

where c_o, α are positive constants.

Inside $S(\rho,\alpha,c_o)$, ξ, k_1, k_2 can grow up to the $O(\frac{1}{\rho^\alpha})$ whereas η and z can grow up to the $O(\frac{1}{\sqrt{\mu}\rho^\alpha})$ and $O(\frac{1}{\sqrt{\varepsilon}\rho^\alpha})$ respectively. Hence for all ξ, k_1, k_2, η and z inside $S(\rho,\alpha,c_o)$, (3.19) can be written as

$$\begin{aligned}\dot{V} \leq & -\frac{a_m}{2}\xi^2 - \frac{\lambda_1}{2}\|\eta\|^2 - \varepsilon\frac{a_2}{2}z^2 - \frac{b_o\sigma_1}{2\gamma_1}(k_1-k_1^*)^2 - \frac{b_o\sigma_2}{2\gamma_2}(k_2-k_2^*)^2 \\ & - \frac{\xi^2}{12}[a_m - \mu\frac{d_1}{\rho^{2\alpha}} - \mu^2\frac{d_2}{\rho^{4\alpha}}] - \frac{\|\eta\|^2}{18}[\lambda_1 - \mu\frac{d_3}{\rho^{2\alpha}} - \sqrt{\mu}\frac{d_4}{\rho^\alpha} - \mu\frac{d_2}{\rho^{4\alpha}}] \\ & - (k_1-k_1^*)^2[\frac{b_o\sigma_1}{2\gamma_1} - \mu^2\frac{d_6}{\rho^{2\alpha}}] - (k_2-k_2^*)^2[\frac{b_o\sigma_2}{2\gamma_2} - \mu^2 d_7] \\ & - \varepsilon z^2[\frac{a_2}{2} - \mu^2\varepsilon\frac{d_8}{\rho^{2\alpha}} - \varepsilon d_9] + \frac{b_o\sigma_1}{2\gamma_1}k_1^{*2} + \frac{b_o\sigma_2}{2\gamma_2}k_2^{*2} + \mu^2 d_{10}\end{aligned} \tag{3.21}$$

where $\lambda_1 = \frac{1}{2}\lambda\min(Q)$ and d_1 to d_{10} are positive finite constants. Choosing $\alpha < \frac{1}{2}$ and $\sigma_1 > O(\rho)$, $\sigma_2 > O(\rho)$, we can find a $\rho^* > 0$ such that for all $\rho \in [0,\rho^*]$

$$\begin{aligned}\dot{V} \leq & -\frac{a_m}{2}\xi^2 - \frac{\lambda_1}{2}\|\eta\|^2 - \varepsilon\frac{a_2}{2}z^2 - \frac{b_o\sigma_1}{2\gamma_1}(k_1 - k_1^*)^2 \\ & - \frac{b_o\sigma_2}{2\gamma_2}(k_2-k_2^*)^2 + \frac{b_o\sigma_1}{2\gamma_1}k_1^{*2} + \frac{b_o\sigma_2}{2\gamma_2}k_2^{*2} + \mu^2 d_{10}\,.\end{aligned} \tag{3.22}$$

Clearly,

$$\dot{V} \leq -\sigma V + \frac{b_o\sigma_1}{2\gamma_1}k_1^{*2} + \frac{b_o\sigma_2}{2\gamma_2}k_2^{*2} + \mu^2 d_{10} \tag{3.23}$$

and therefore V is bounded and (3.17) holds.

IV. SIMULATION RESULTS

The effectiveness of the first order adaptive controller (3.2) to (3.4) is tested by simulating the overall nonlinear closed loop system (2.1) to (2.6) and (3.2) to (3.4) on a digital computer. The reference model is chosen to be

$$\dot{y}_m = -2y_m + 2r \tag{4.1}$$

and the adaptive gains are chosen as $\gamma_1 = \gamma_2 = 5$.

The time response of the output (daily production) y is shown in Fig. 2 for $r = 1.2$ and $\sigma_1 = \sigma_2 = 0.01$. The output converges to the desired output with decreasing small oscillations. The convergence rate is rather slow compared to the one obtained in [1] using a self-tuning regulator. This is probably due to the fact that in [1] no reference model was used, whereas in our case the convergence of the scheme is affected by the choice of (3.1).

Furthermore, the choice of γ_1,γ_2 has also an impact on the convergence rate and several runs are required in order to choose adaptive gains giving reasonable convergence rates. The choice of σ_1 and σ_2 affected the damping of the system very much. By increasing σ_1,σ_2 we get less or even no oscillations, but the steady state error increases. Low σ_1,σ_2 give high oscillations and a lower rate of convergence.

The scheme is also tested for a sinusoidal reference input $r = 1.2 + 0.5 \sin 2t$ and the output y response is plotted in Fig. 3. In this case the choice of γ_1,γ_2 and σ_1,σ_2 is also found to affect the convergence rates and steady-state error.

V. CONCLUSIONS

A first order adaptive controller designed for a simplified linearized model of a 5th order fermentation process is analyzed. Bounds for stability are obtained for the linearized 5th order model. Simulation results for the overall closed-loop nonlinear system show that the simple control design is effective for regulation and tracking. In the case of regulation, however, the convergence rate of the output is lower than that obtained using a self-tuning regulator. Further analysis and study is required in order to show stability for the overall closed-loop system and improve the convergence of the scheme by a proper selection of the design parameters.

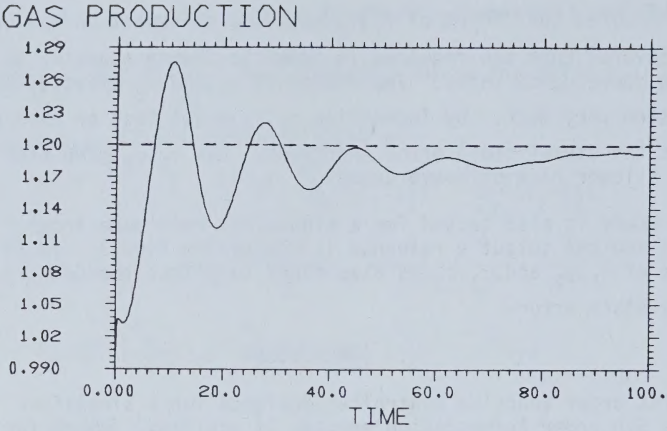


Fig. 2. Time response of the output (daily gas production) for r = 1.2.

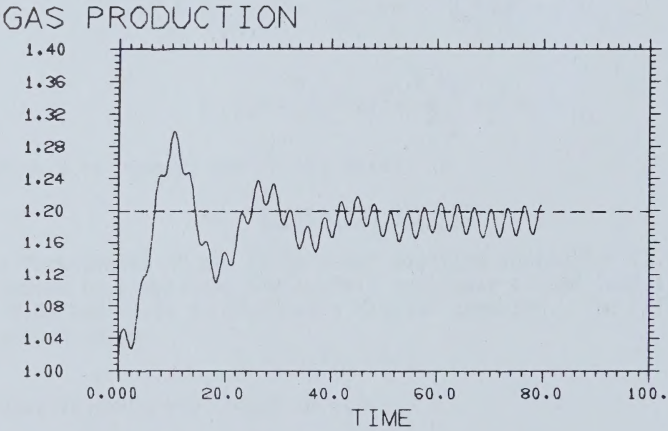


Fig. 3. Time response of the output (daily gas production) for r = 1.2 + 0.5 sin 2t.

VI. REFERENCES

[1] G. Bastin, D. Dochain, M. Haest, M. Installe, Ph. Opdenacker, "Modeling and Adaptive Control of a Continuous Anaerobic Fermentation Process," IFAC Workshop on Modeling and Control of Biotechnical Systems, Helsinki, August 1982.

[2] D.T. Hill and C.L. Barth, "A Dynamic Model for Simulation of Animal Waste Digestion," Journal W.P.C.F., October 1977.

[3] K.S. Narendra and L.S. Valavani, "Stable Adaptive Controller Design-Direct Control," IEEE Trans. on Automatic Control, Vol. AC-24, No. 4, pp. 570-583, August 1978.

[4] P.A. Ioannou and P.V. Kokotovic, "Adaptive Systems with Reduced Models," Springer-Verlag, Vol. 47, 1983.

[5] P.A. Ioannou and P.V. Kokotovic, "Robust Redesign of Adaptive Control," to appear in IEEE Trans. on Automatic Control, Vol. 29, No. 3, March 1984.

[6] P.A. Ioannou, "Robust Redesign of Adaptive Control in the Presence of Disturbances and Unmodeled Dynamics," IFAC Workshop on Adaptive Systems in Control and Signal Processing, June 1983, San Francisco.

[7] P.A. Ioannou and P.V. Kokotovic, "Improvement of Robustness of Adaptive Schemes," Proceedings of the Third Yale Workshop on Applications of Adaptive Systems Theory, June 15-17, 1983.

THE STATISTICAL PROPERTIES OF THE RESIDUAL IN SYSTEM IDENTIFICATION

J.M. FLENTZ, Y.F. HUANG, W.J. GAJDA, JR.
University of Notre Dame
Notre Dame, IN 46556

ABSTRACT

The Suen-Liu method of system identification, used to determine structural parameters of an autoregressive moving average system, has been extended to permit complete automation for implementation. Simplifying assumptions, pertinent to physical systems, permit rapid implementation of the new algorithm, which involves techniques of statistical inference. The primary purpose of considering this problem is to efficiently implement the system identification technique in order to detect and predict frost or ice forming on the power rail in cold weather transportation systems.

I. INTRODUCTION

A problem that commonly occurs in cities which are subject to freezing weather, and which have electric mass-transit systems, is the formation of frost and ice on the power rail. The frost/ice causes interruptions in the required current, which in turn causes slowing or even stopping of the train. In an attempt to solve this problem, many methods are being considered to detect and possibly predict the formation of frost and ice. These detection or prediction methodologies would eventually result in cost-effective use of track heaters, chemicals, or other ice-removal techniques.

A method of detection being considered is the monitoring of the drawn current from the power rail, and hence the degree of its interruption. The monitored data could then be analyzed, in some way, along with an appropriate interpretation of the results. The important issue then is the selection of a suitable method of analysis of the data. For the considered application, the method should be as autonomous as possible. Simplicity is also a requirement. While spectral-analysis methods are being studied at the time of this writing, a suitable method of system identification has been found.

For this study, it is assumed that the current-interruption process can be modeled by an autoregressive moving-average (ARMA) equation. As it is also known that the presence of ice affects this process, it then becomes appropriate to search for any changes in the ARMA equation.

The method of system identification developed by Suen and Liu ([1],[2]), hereafter referred to as the "Suen-Liu method," uses least-squares methods to determine the structural parameters, namely, the order, the degree of noncausality, and the coefficients of an ARMA model for a process. The ability of the method to determine structural parameters is important in the design and implementation of efficient, accurate, and cost-effective techniques for ice-removal in cold weather transit systems. The result of the Suen-Liu method is a graph from which the structural parameters for a minimal-order system describing the data can be determined. While the extraordinary simplicity of the final result must be noted, its interpretation and resulting decision-making require human intervention. Thus, in order to fully automate the process of identification of structural parameters, and to provide a better statistical justification for the decision-making procedure, analysis has been performed to extend the Suen-Liu method. This involves an application of the general theory of statistical inference. Specifically, the concept of the confidence interval will be employed.

This paper is summarized in the following. After a brief literature review, Section II, the problem formulation and a summary of Suen-Liu results are presented. Operation of the Suen-Liu method is described in conjunction with the above discussion. It is then shown in Section III that assumptions of Gaussian inputs and observation noise result in a χ^2-distributed residual. This, in turn, leads to the discussion (in Section IV) of the main result of this paper: the formulation and implementation of the techniques of statistical inference for the determination of structural parameters of the process under consideration. The paper concludes with Section V, which contains preliminary results of other pertinent topics being investigated.

II. PRELIMINARY DISCUSSION

Literature Review

Many system-identification algorithms exist which could be applied to this problem. One possibility is the Steiglitz-McBride algorithm [3] or its extension [4]. However, this iterative algorithm can, occasionally, become unstable as addressed more recently in [5]. Another possibility is the maximum-likelihood method, which, based on assumptions to be given later, could be applied to this problem. However, some disadvantages of the method are discussed in [6]. The solutions presented therein require extra computation to test the validity of a result. Still more identification techniques are discussed in a very comprehensive and useful survey paper [7]. In that paper, a variant of the Suen-Liu method is discussed. That method involves the determination of an F-distributed random variable that tests the significance of the reduction of the square of the norm of the least-squares residual vector. The disadvantage of that method is that, for each estimate of system order, a search must be performed to evaluate the minimum value of the above norm-squared quantity. In [8], a comparison is made between various methods of determination of system order. The most notable method described therein is Akaike's Information Criterion. The only disadvantage is that this procedure may not yield a minimal system order, as mentioned in [1]. Minimality could be an important factor in the ice-detection problem, since it affects the efficiency of implementation in practice. Some new methods ([9],[10]) are being studied at the time of this writing. In [9], information-matrix techniques are used, while [10] is restricted to autoregressive processes.

Among the papers for the literature review, [11] describes a procedure that appears to be a generalization of the Suen-Liu method. In that paper, an ARMA model with colored observation noise is considered. In the Suen-Liu method (and in the present paper), the observation noise is assumed to be white. The manner in which the observation noise appears in the ARMA equation is described in the next section, where the Suen-Liu method is summarized.

Problem Formulation and Summary of the Suen-Liu Method

In this paper, it is assumed that the current-interruption process is modeled by the ARMA equation

$$y_k + \sum_{i=1}^{n} a_i y_{k-i} = \sum_{i=-L}^{n} b_i u_{k-i} + v_k \tag{1}$$

$\{y_k\}$ is the observed output sequence of the system, $\{u_k\}$ is its input sequence, and $\{v_k\}$ is the observation noise sequence. The Suen-Liu method determines n, the order, L, the degree of noncausality, and the $\{a_i\}$ and $\{b_i\}$ of Equation (1).

Suppose that an input-output data sequence of length K is obtained. The purpose here is to estimate n and L. Then, $y[n+1,K-L]$, $(K-L-n)$ dimensional vector which consists of observed outputs from time n+1 to time K-L, is formed. Also formed is a $(2n+L+1) \times (K-L-n)$ matrix consisting of observed inputs and outputs. This matrix may sometimes be referred to as a Hankel matrix. The basis for this is simply the rearrangement of Equation (1). The results are

$$\begin{bmatrix} y_{n+1} \\ \cdot \\ \cdot \\ \cdot \\ y_{K-L} \end{bmatrix}^T = \begin{bmatrix} y_n & y_{n-1} & \cdots & y_1 & u_{n+1+L} & u_{n+1} & \cdots & u_1 \\ \cdot & \cdot & & \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot & \cdot & \cdot & & \cdot \\ y_{K-L-1} & y_{K-L-2} & \cdots & y_{K-L-n} & u_K & u_{K-1} & \cdots & u_{K-L-n} \end{bmatrix}^T \times$$

$$\begin{bmatrix} -a_1 \\ -a_2 \\ \cdot \\ \cdot \\ \cdot \\ -a_n \\ b_{-L} \\ b_{1-} \\ \cdot \\ \cdot \\ \cdot \\ b_n \end{bmatrix}^T + \begin{bmatrix} v_{n+1} \\ \cdot \\ \cdot \\ \cdot \\ v_{K-L} \end{bmatrix}^T \underline{\triangle}\ W^T(n,L)\ \theta^T(n,L) + v^T[n+1,K-L] \tag{2}$$

The Suen-Liu method then uses the familiar least-squares result which leads to the estimate of θ, i.e., $\hat{\theta} = yW^+$, where $\hat{\theta}$ is the least-squares estimate of θ, and W^+ is the pseudoinverse of W. One can then obtain the estimate of y, given by $\hat{y} = y\,W^+W$. Let the residual vector be defined by

$$\begin{aligned} r[n+1,K-L] &= y[n+1,K-L] - \hat{y}[n+1,K-L] \\ &= y[n+1,K-L](I-W^+(n,L)W(n,L)) \end{aligned} \tag{3}$$

Let e(n,L) be defined as follows:

$$\begin{aligned} e(n,L) &= r[n+1,K-L]\ r^T[n+1,K-L] \\ &= y[n+1,K-L](I-W^+(n,L)W(n,L))y^T[n+1,K-L] \end{aligned} \tag{4}$$

The last equality follows from the idempotence of $I-W^{+}(n,L)W(n,L)$. Substitution of Equation (2) into Equation (4) yields an expression which, with the aid of a lemma in [1], yields

$$e(n,L) = v[n+1,K-L](I-W^{+}(n,L)W(n,L))\, v^{T}[n+1,K-L] \tag{5}$$

Equation (4) has the practical value necessary for implementation of the Suen-Liu method. Equation (5) aids in the study of the statistical properties of the residual, and it is discussed further in Section III.

The next step is, for the same batch of data, to generate values of e(n,L) for estimates of n and L. These values, however, can be quite large in practice, and hence a quantity called the normalized residual is used to avoid annoyingly extreme values. The normalized residual is defined as

$$\varepsilon(n,L) = e(n,L)/\|y[n+1,K-L]\|^{2} \tag{6}$$

Implementation of the Suen-Liu method requires plots of $\varepsilon(n,L)$ versus n, with L as a parameter. The final result is a family of curves which, at values of n and L greater than or equal to the true n and L, become approximately equal to a common (constant) value. Thus, implementation of the Suen-Liu method can be summarized as follows:

> The true values of n and L, respectively denoted as n° and L°, are the first values of n and L for which the normalized-residual curves attain an approximately common, constant value.

Obviously, determination of the system order here is based on a visual inspection of the curves.

Results of two examples of implementation of the method are now presented. In each case, records of input and output data are generated in conjunction with arbitrarily selected ARMA models. The $\{u_k\}$ is i.i.d. (independent and identically distributed) with a Gaussian distribution sequence N(0,10). Observation noise is also considered in the examples. For each case, $\{v_k\}$ is i.i.d. with a Gaussian distribution N(0,1). Also, $\{u_k\}$ and $\{v_k\}$ are uncorrelated.

The original purpose of these examples is to examine the feasibility of the Suen-Liu method to our problem here. A program which employs LINPACK subroutines [12] is described in [13]. According to [2], the variance of $\{v_k\}$ should be small with respect to the variance of $\{y_k\}$, to ensure accurate results for $\hat{\theta}(n,L)$.

For both examples, 100 independent groups of 100 input/output pairs were generated. For particular estimates of n and L, each group was used to generate an element of a sequence of $\varepsilon(n,L)$ and an element of a vector sequence of $\hat{\theta}(n,L)$. Both sequences were averaged, and the results of this averaging were used in the final determination of n°, L°, and associated ARMA coefficients.

The first example has n°=3 and L°=1. The ARMA equation used to generate appropriate data is

$$y_k+0.1y_{k-1}-0.05y_{k-2}+0.15y_{k-3}=u_{k+1}+0.7u_k-0.2u_{k-1}+0.7u_{k-2}-1.2u_{k-3}+v_k \tag{7}$$

Estimates of n (L) range from 1 to 6 (from 0 to 3). Results are shown in both Figure 1 and Table 1.

The second example assumes n°=2 and L°=0. The ARMA equation used to generate appropriate data is

$$y_k+0.1y_{k-1}-0.05y_{k-2} = 0.7u_k-0.2u_{k-1}+0.7u_{k-2}+v_k \tag{8}$$

Estimates of n and L have the same range as the first example. Results are shown in both Figure 2 and Table 2.

III. ANALYSIS

This section presents assumptions and describes an important proposition that is to be used in Part IV.

Assumptions

The assumptions for the analysis here are described in the following

1. $\{u_k\}$ is an i.i.d. $N(0,\sigma_u^2)$ sequence
2. $\{v_k\}$ is an i.i.d. $N(0,\sigma_v^2)$ sequence
3. $\{u_k\}$ and $\{v_k\}$ are uncorrelated
4. $\sigma_u^2 >> \sigma_v^2$

While the above assumptions have immense simplification of the analysis, they are compatible with the physical process of current-interrution. Furthermore, if many processes (assumed independent) contribute to this process, then the central-limit-theorem suggests the rationality of these assumptions. The statistics of $\{v_k\}$ are sufficient, however, to aid in the proof of a useful proposition.

Density of e(n,L)

For reasons to be explained in Part IV, it is convenient to determine the probability density function (pdf) of e(n,L) first. The pdf is presented in the following proposition.

Proposition: Under the assumption of $\{v_k\}\sim N(0,\sigma_v^2 I)$, e(n,L) has a χ^2 distribution.

Proof: The proof can begin with Equation (5), along with the fact that $I-W^+W$ is idempotent. According to [2], the rank of this idempotent matrix is K-3n-2L-1. Using a result available in [14], e(n,L) has a χ^2 distribution, with K-3n-2L-1 degrees of freedom. By [15], the explicit form of the pdf of e(n,L) is

$$p_E(e(n,L)) = \frac{e(n,L)^{(N-2)/2}}{2^{N/2}\ \sigma_v^N\ \Gamma(N/2)} \exp[-e(n,L)/2\sigma_v^2] \tag{9}$$

for $e(n,L) \geqslant 0$. In Equation (9), $N \triangleq K-3n-2L-1$. Q.E.D.

Knowledge of the pdf is very useful in verifying other statistical results of e(n,L) [1]. Presently, this leads to the main result of this paper.

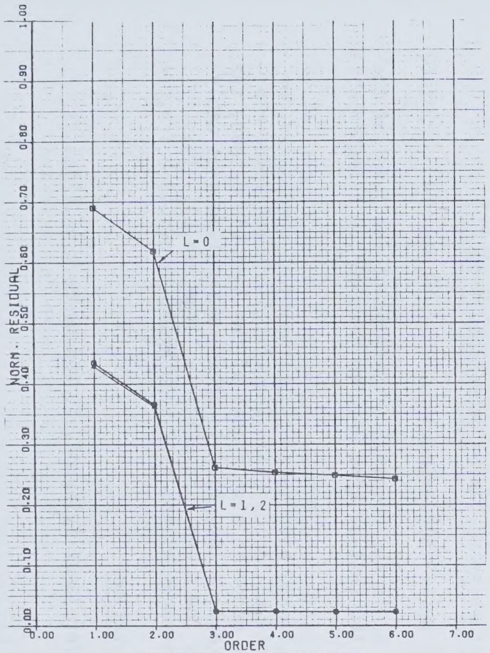


FIGURE 1: GRAPHICAL RESULTS FOR THIRD-ORDER, 1-STEP NONCAUSAL EXAMPLE

TABLE 1: ESTIMATED ARMA COEFFICIENTS FOR THIRD-ORDER, 1-STEP NONCAUSAL EXAMPLE

COEFFICIENT	VALUE
A_1	0.0969
A_2	-0.0484
A_3	0.1483
B_{-1}	1.0
B_0	0.6924
B_1	-0.1993
B_2	0.7003
B_3	-1.2058

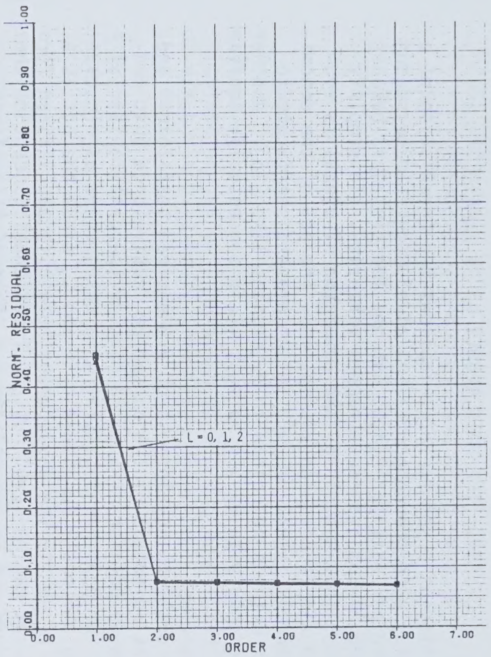


FIGURE 2: GRAPHICAL RESULTS FOR SECOND-ORDER CAUSAL EXAMPLE

TABLE 2: ESTIMATED ARMA COEFFICIENTS FOR SECOND-ORDER CAUSAL EXAMPLE

COEFFICIENT	VALUE
A_1	0.1062
A_2	-0.0418
B_0	0.6969
B_1	-0.1949
B_2	0.7062

IV. THE MAIN RESULT

The availability of a pdf for $e(n,L)$ allows the formulation of a simple test to fully automate determination of n° and L°. The reason for determining the pdf only for $e(n,L)$ but not for $\varepsilon(n,L)$ is because $\varepsilon(n,L)$ is the ratio of two dependent random variables. To simplify the discussion here, only the pdf of the numerator of $\varepsilon(n,L)$ is evaluated. Even with simple Gaussian assumptions, the pdf of the denominator of $\varepsilon(n,L)$, namely $\|y[n+1,K-L]\|^2$, is, at present, difficult to determine with the methods described in references like [15]. Another reason for working only with $e(n,L)$ is that it is conceptually equivalent to $\varepsilon(n,L)$.

This section outlines development of a one-sided test for n° and L°. The test takes advantage of the conceptual equivalence just mentioned. With a known pdf for $e(n,L)$, thresholds (percentile levels) can be determined for hypothesis-testing of a given value of $e(n,L)$. The thresholds are obviously functions of n and L, since the pdf is a function of n and L. While the development herein is specialized for this distribution, it can be seen that the fundamental concept of the inferencing procedure can be generalized for other distributions.

Formulation

Formulation of the hypothesis test is straight-forward [16]. The test is binary:

H_0: $(n,L) < (n^\circ,L^\circ)$

H_1: $(n,L) \geq (n^\circ,L^\circ)$

The design begins by selecting the level of the test, which is a fraction. The fraction is the probability that $e(n,L)$ is less than the threshold, $T(n,L)$. Thresholds can be determined from

$$\text{LEVEL} = \int_0^{T(n,L)} p_E(e(n,L))\, de(n,L) \tag{10}$$

In other words, $T(n,L)$ is the percentile function of the level of the test. The level of the test of this problem here can be selected as 0.95 or 0.99.

A particular n,L, and associated $\theta(n,L)$ would be candidates for the true values if $e(n,L)$ is less than $T(n,L)$. Since obtaining minimality of the system order is also a goal of the Suen-Liu method, the null hypothesis is $(n,L) < (n^\circ,L^\circ)$. In terms of least-squares analysis, an inadequately low n and L result in a large $e(n,L)$. This results in the restatement of the null hypothesis as $e(n,L) > T(n,L)$.

Algorithmic Implementation

The algorithm for identifying n°,L°, and hence $\theta(n^\circ,L^\circ)$ is therefore:

```
1. Set $L = 0$
2. Set $n=1$
3. Calculate $e(n,L)$
4. If $e(n,L) \leq T(n,L)$ go to Step 7
```

5. If $n=n_{MAX}$ go to Step 6
   Otherwise, set $n=n+1$ and go to Step 3
6. If $L=L_{MAX}$ go to Step 8
   Otherwise, set $L=L+1$ and go to Step 2
7. Signify n and L as n° and L°; stop
8. Signify that $n° > n_{MAX}$ and L° may be greater than L_{MAX}; increase n_{MAX} and, if desired, L_{MAX}, and repeat algorithm.

Some comments are in order for use of the algorithm:

- This algorithm makes use of the same batch or batches of data to calculate e(n,L) or a sequence of e(n,L), respectively. If a sequence of e(n,L) is generated, then the mean of the sequence is used as the test statistic.

- Values of T(n,L) can either be stored in computer memory or calculated via an appropriate quantile function.

- If no conclusion can be reached for (n°,L°), n_{MAX}, rather than L_{MAX}, should be increased first. Repeat the algorithm for values of n greater than the previous n_{MAX}.

Figure 3 is a graphical summarization of the decision involved in the algorithm. It shows the position of e(n,L) relative to the confidence interval [0,T(n,L)] for values of (n,L) relative to (n°,L°).

	$n<n_0$	$n=n_0$	$n>n_0$
$L<L_0$	OUTSIDE	OUTSIDE	OUTSIDE
$L=L_0$	OUTSIDE	INSIDE	INSIDE
$L>L_0$	OUTSIDE	INSIDE	INSIDE

Figure 3: Position of e(n,L) W.R.T. [0,T(n,L)]

Applying the χ^2-result of Equation (9) to the algorithm, it is noted that:

- T(n,L) comes from readily-available tables; a value of T(n,L) is the appropriate percentile value of a χ^2 distribution having K-3n-2L-1 degrees of freedom.

- If the true value of σ_v^2 is unknown, it can be calculated from

$$\sigma_v^2 = \sqrt{\frac{\sigma_\varepsilon^2}{2N}} \qquad (11)$$

  where σ_ε^2 is the unbiased variance of e(n,L) obtained from experimental data, and N=K-3n-2L-1.

This identification algorithm was implemented and tested by using the two example plants of Section II. Results were successful.

Non-Unity Observation-Noise Variance

The implementation of the decision algorithm for the examples made use of a χ^2 quantile table that was available. It should be remarked here that the analysis and simulation here correspond to a unity observation-noise variance. If this variance is other than one, the test statistic must be transformed as

$$x \;\underline{\underline{\Delta}}\; e(n,L)/\sigma_v^2 \tag{12}$$

and then tested against the available threshold.

V. CONCLUSION

A method has been presented which extends and automates the Suen-Liu method of ARMA system identification. Its purpose is to determine parameters governing current-interruption processes in electric mass-transit systems. As the considered process is physical in nature, the input and observation noise are assumed to be Gaussian. These assumptions simplify analysis, and lead to an algorithm that is easily implemented. The implementation can be used to modify programs such as the one discussed in [12].

The following topics are currently being considered:

- Other distributions for $\{u_k\}$ and $\{v_k\}$
- pdf for $\varepsilon(n,L)$

For the first issue, development up to and including Equation (5) is applicable. In [1], it is mentioned that if $\{v_k\}$ is i.i.d. with variance σ_v^2, then the expected value of Equation (5) would be $(K-3n-2L-1)\sigma_v^2$. While this result applies to all density functions, it serves only as a check for what may be a cumbersome analysis.

An expression for the pdf of $\varepsilon(n,L)$ is still being sought. As stated in Part IV, it would be very difficult to do so with classical methods. However, some histographic results have been obtained for the plant with $n°=3$, $L°=1$. Preliminary inspection indicates that the apparent standard deviation of the normalized residual decreases substantially for $n \geqslant n°$, even when $L<L°$. These visual observations are all that have been obtained at the present. Analysis is clearly required to determine the cause of this behavior, and hence a possible result for the pdf of $e(n,L)$.

An expression is also being sought for the pdf of the square of the norm of a Gaussian random vector having a non-diagonal covariance matrix. This problem arose while considering $\|y[n+1,K-L]\|^2$, the denominator of $\varepsilon(n,L)$. Again, the classical methods become intractable, especially as the number of elements in $y[n+1,K-L]$ are on the order of 100.

To say the least, the above areas are challenging, and their elegant resolution would be most welcome.

VI. ACKNOWLEDGEMENT

The authors are deeply grateful to Dr. Ruey-wen Liu, whose guidance with implementation of the Suen-Liu method is considered essential. This work has been supported by the Department of Transportation, Urban Mass Transportation Administration, under Grant IN-06-0009 titled Cold Weather Transit Technology Program. This is Report No. CWTT-22-005.

REFERENCES

[1] Suen, L.C. and R. Liu, "Determination of the Structure of Multivariable Stochastic Systems," IEEE Transactions on Automatic Control, Volume AC-23, Number 3, June 1978.

[2] Suen, L.C. and R. Liu, "A Normalized Residual Criterion for the Determination of Structure of Linear Systems", Journal of Policy Analysis and Information Systems, Volume 1, July 1977.

[3] Steiglitz, K. and L.E. McBride, "A Technique for the Identification of Linear Systems," IEEE Transactions on Automatic Control, Volume AC-10, Number 5, October 1965.

[4] Stefanski, A. and C.N. Weygardt, "Extension of the Steiglitz and McBridge Identification Technique," IEEE Transactions on Automatic Control, Volume AC-16, Number 5, October 1971.

[5] Stoica, P. and T. Söderström, "The Steiglitz-McBride Identification Algorithm Revisited - Convergence Analysis and Accuracy Aspects," IEEE Transactions on Automatic Control, Volume AC-26, Number 3, June 1981.

[6] Bohlin, T., "On the Problem of Ambiguities in Maximum Likelihood Identification," Automatica, Volume 7, p. 199, 1971.

[7] Aström, K. J. and P. Eykhoff, "System Identification--A Survey," Automatica, Volume 7, p. 123, 1971.

[8] Söderström, T., "On Model Structure Testing in System Identification," International Journal of Control, Volume 26, p. 1, 1977.

[9] Guidorzi, R.P., M.P. Losito and T. Muratori, "The Range Error Test in the Structural Identification of Linear Multivariable Systems," IEEE Transactions on Automatic Control, Volume AC-27, Number 5, October 1982.

[10] Baram, Y., "Recovering the Poles of Linear Systems," IEEE Transactions on Automatic Control, Volume AC-27, Number 6, December 1982.

[11] van den Boom, A.J.W. and A.W.M. van den Enden, "The Determination of the Order of Process and Noise Dynamics," Preprints of the Third IFAC Symposium on Identification and Parameter Estimation, 1973.

[12] Dongarra, J.J., C.B. Moler, J.R. Bunch, and G.W. Stewart, LINPACK Users' Guide, SIAM, Philadelphia, 1979.

[13] Flentz, J.M., "Users' Manual for Suen-Liu Computer Program," Report No. CWTT-22-003, March 1983.

[14] Goodwin, G.C. and R.L. Payne, Dynamic System Identification: Experiment Design and Data Analysis, Academic Press, New York, 1977.

[15] Papoulis, A., Probability, Random Variables, and Stochastic Processes, McGraw-Hill, New York, 1965.

[16] Bickel, P.J. and K.A. Doksum, Mathematical Statistics: Basic Ideas and Selected Topics, Holden-Day, San Francisco, 1977.

CORRECTIVE METHODS FOR IDENTIFICATION IN THE CLOSED INFERENTIAL CONTROL LOOP

CHENG WU
Tsinghua University
Dept. of Automation
Beijing
People's Republic of China

JANOS GERTLER
Visiting Professor
Case Western Reserve University
Systems Engineering Department
Cleveland, Ohio 44106

ABSTRACT

Two heuristic algorithms are introduced for the identification of linear discrete processes in closed loop, with known controller. One algorithm is an adaptation of the instrumental variable principle; it can be used, with any controller, if the output noise is a zero mean random sequence. The other algorithm is an iterative least square procedure developed for step-disturbance; this can only be applied if the controller (or the process) is integrating. In principle, both methods give consistent estimates under the respective assumptions. Simulation studies have verified the convergence of the algorithms and their superior accuracy over the ordinary least squares.

INTRODUCTION

Inferential control is a special closed-loop scheme that makes explicit use of the model of the controlled process [1]. If the model is a perfect replica of the process the performance of the closed loop can be exactly prescribed. In practice this is seldom the case. The performance of the loop may be kept close to ideal if the process is periodically identified and the model updated accordingly.

An effort is under way to develop closed-loop identification methods and test them in the inferential control scheme [3]. Some of these methods are indirect variants of the single-shot least-square technique [2]. As can be expected, such tricks do not result in significant improvement over the direct least squares [4]. Other methods are of corrective nature taking into account the noise and its feed-back in the closed loop by explicit corrections (filtering). This approach leads to a more complex iterative algorithm and considerably better accuracy.

In this paper, two heuristic corrective closed-loop identification methods are described, both suitable for inferential control under different conditions. One is an adaptation of the instrumental variable principle; it is applicalbe to any closed loop with zero-mean random output disturbance. The other method is an iterative least-square procedure developed for step-function output disturbance; it can only be applied if there is an integrator in the loop. This latter condition is alwyas satisfied with inferential control. The viability of both methods has been verified by computer simulation.

The main properties of inferential control are summarized in the Appendix. The body of the paper starts with a brief introduction of least squares and instrumental variables. This is followed by the description of the two methods. Finally, some simulation results are presented.

LEAST SQUARES AND INSTRUMENTAL VARIABLES

Consider the single-input single output linear time-invariant process of Figure 1 with input u(t), output y(t) and noise v(t). The process is described by the discrete input-output relationship

$$y(t) = G_p(z)u(t) + v(t) \qquad (1)$$

where

$$G_p(z) = \frac{B_p(z)}{1+A_p(z)} = \frac{b_0 + b_1 z^{-1} + \ldots + b_n z^{n}}{1+a_1 z^{-1} + \ldots + a_n z^{n}} \qquad (2)$$

is the discrete transfer function.

An equivalent form of the input-output relationship is

$$y(t) = B_p(z)\, u(t) - A_p(z)y(t) + [1 + A_p(z)]\, v(t) \qquad (3)$$

or $\quad y(t) = \underline{\underline{x}}^T(t)\underline{p} + e(t) \qquad (4)$

where

$$\underline{p} = [b_0, \ldots, b_n, a_1, \ldots, a_n] \qquad (5)$$

is the parameter-vector,

$$\underline{\underline{x}}^T(t) = [u(t), \ldots, u(t-n\Delta t), -y(t-\Delta t), \ldots, -y(t-n\Delta t)] \qquad (6)$$

is the vector of observations at time t (and Δt is the sampling interval) and

$$e(t) = [1 + A_p(z)]\, v(t) \qquad (7)$$

is the equation error at time t.

The least-square estimate $\hat{\underline{p}}$ of the parameter vector can be obtained by the well known formula

$$\hat{\underline{p}} = (\underline{\underline{X}}^T\underline{\underline{X}})^{-1}\, \underline{\underline{X}}^T\underline{y} \qquad (8)$$

where $\underline{\underline{X}}$ is a matrix formed of a number of consecutive observation vectors $\underline{\underline{x}}^T(t)$ and $\underline{y}$ is a vector formed of a number of such observations of the output y(t).

Substituting the vector-form of Eqn. (4) into (8) yields

$$\underline{p} = (\underline{\underline{X}}^T\underline{\underline{X}})^{-1}\, \underline{\underline{X}}^T(\underline{\underline{X}}\underline{p} + \underline{e}) = \underline{p} + (\underline{\underline{X}}^T\underline{\underline{X}})^{-1}\, \underline{\underline{X}}^T\underline{e} \qquad (9)$$

where the second term is the estimation error. The estimates can be made consistent, that is the error term zero as the number of observations tends to infinity, if

- the expectation of e(t) is zero, and
- $\underline{\underline{x}}^T(t)$ is uncorrelated with e(t).

The equation error e(t) depends only on the noise v(t), see Eqn. (7). If v(t) is of zero mean, so is e(t). However, $\underline{\underline{x}}(t)$ is usually not uncorrelated with e(t), since

- $y(t-k\Delta t)$, depends on $e(t-k\Delta t)$, $e(t-k\Delta t-\Delta t)$ etc. and thus it is correlated with e(t), unless e(t) is a white sequence,
- $u(t-k\Delta t)$ may also depend on $e(t-k\Delta t)$, etc. if the process operates in closed loop.

The basic idea of the instrumental variable approach [5] is to partially replace matrix $\underline{\underline{X}}$ in Eqn. (8) by an "instrumental variable" matrix $\underline{\underline{X}}_0$. This matrix is selected in such a way that

- $\underline{\underline{X}}_0^T$ is uncorrelated with $\underline{\underline{e}}(t)$, and
- $\underline{\underline{X}}_0^T\underline{\underline{X}}$ is invertible.

With such substitution the equation

$$\hat{\underline{\underline{p}}} = (\underline{\underline{X}}_0^T\underline{\underline{X}})^{-1}\ \underline{\underline{X}}_0^T\underline{\underline{y}} \qquad (10)$$

can be evaluated and it yields consistent estimates.

INSTRUMENTAL VARIABLES IN CLOSED LOOP

Consider now the closed-loop situation shown in Figure 2. Here $G_c(z)$ is a known controller transfer function

$$G_c(z) = \frac{B_c(z)}{1+A_c(z)} = \frac{b_{c1}z^{-1}+\ldots+b_{ck}z^{-k}}{1+a_{c1}z^{-1}+\ldots+a_{ck}z^{-k}} \qquad (11)$$

The system output y(t) and the process input u(t) are observed. It is assumed that the noise v(t) is a zero-mean random sequence, independent of the system input (setpoint) w(t).

Obviously, the observations $\underline{\underline{x}}(t)$ are not uncorrelated with the equation error e(t) since both the output y(t) and the process input u(t) depend on the noise v(t). Thus the least-square estimates are inconsistent. One possibility of attaining consistent estimates is constructing an instrumental variable matrix through the separation of the observations into noise-dependent and setpoint-dependent parts. By superposition

$$\underline{\underline{x}}(t) = \underline{\underline{x}}_w(t) + \underline{\underline{x}}_v(t) \qquad (12)$$

where $\underline{\underline{x}}_w(t)$ is the setpoint-dependent and $\underline{\underline{x}}_v(t)$ is the noise-dependent component. The setpoint-dependent component is uncorrelated with the noise so $\underline{\underline{X}}_w$ would qualify as an instrumental variable matrix. That is, consistent estimates can be obtained from the formula

$$\hat{\underline{\underline{p}}} = (\underline{\underline{X}}_w^T\underline{\underline{X}})^{-1}\ \underline{\underline{X}}_w^T\underline{\underline{y}} \qquad (13)$$

To perform the separation of the observations one needs the transfer functions $G_c(z)$ and $G_p(z)$. Since $G_p(z)$ is not known, an iterative procedure is necessary: first the process parameters are estimated by least squares, then this estimate is used in a first separation, leading to a new estimation, improved separation, etc.

The algorithm can be implemented off-line, iterating on one batch of observations or semi on-line, collecting several batches of observations and performing one or a number of iterations on each batch. In principle, it is also possible to make the algorithm completely on-line (recursive), performing a new iteration following each sample; this approach, however, is computationally impractical.

The convergence of the algorithm has not been proved so far. Simulation results, to be reported later in this paper, have verified its good properties.

The Separation Procedure

The separation procedure is based on simple block-operations. First the actual values of the noise are estimated as

$$\hat{v}(t) = y(t) - \hat{G}_p(z)\, u(t) \tag{14}$$

where $\hat{G}_p(z)$ is the latest estimate of $G_p(z)$. Then the estimates of $u_w(t)$ and $y_w(t)$, the setpoint-dependent components of $u(t)$ and $y(t)$, are obtained, making use of Eqn. (12) and the respective closed-loop transfer functions, as

$$\hat{u}_w(t) = \frac{u(t) + G_c(z)\, y(t)}{1 + G_c(z)\, \hat{G}_p(z)} \tag{15}$$

$$\hat{y}_w(t) = \hat{G}_p(z)\, \hat{u}_w(t) \tag{16}$$

With inferential control (see the Appendix), the structure of the closed loop is more complex. Making use of the equivalent non-inferential controller as described in Eqn. (A-8), Equation (15) is replaced by

$$\hat{u}_w(t) = \frac{G_M(z)[1-G_F(z)]u(t) + G_F(z)\, y(t)}{\hat{G}_p(z)\, G_F(z) + G_M(z)[1-G_F(z)]} \tag{17}$$

Equation (16) is unchanged.

MODIFIED LEAST-SQUARES IN INTEGRATING CLOSED-LOOPS

In many industrial situations, the output distrubance v(t) is a set of random step-changes rather than a zero-mean random sequence. Thus the assumption that the equation error e(t) has zero mean is not valid and the instrumental variable method would not yield consistent estimates.

In this case, a possible approach is to base the least-square estimation entirely on the setpoint-dependent part of the observations:

$$\hat{\underline{p}} = (\underline{\underline{X}}_w^T \underline{\underline{X}}_w)^{-1}\, \underline{\underline{X}}_w^T \underline{y}_w \tag{18}$$

This estimate, of course, is consistent. However, the separation again requires iteration and is more difficult than in case of instrumental variables.

In the previous separation algorithm $\hat{y}_w(t)$ was obtained from $\hat{u}_w(t)$ with the last estimate $\hat{G}_p(z)$, see Eqn. (16). With this approach, Eqn. (18) would return the first estimate of $G_p(z)$ in each iteration step. To avoid this, $\hat{u}_w(t)$ and $\hat{y}_w(t)$ must be obtained separately from each other. This can be done if

- there is at least one integrator in the loop
- the setpoint w(t) can be observed instead of the process input u(t)

The presence of an integrator means that the noise-dependent component

$y_v(t)$ is zero in steady state. That is, the setpoint-dependent component $y_w(t)$ can be approximated as $\hat{y}_w(t) \approx y(t)$. On the other hand, the setpoint dependent part of the process input, $u_w(t)$, can be estimated from the setpoint, making use of the latest estimate of the process transfer function.

In the iterative procedure, estimation and separation alternate, just like in the instrumental variable approach; the first estimate is an ordinary least squares. If the scheme is inferential, convergence can be improved by a two-level iteration. The noise-dependent component is present in the output during the transients following each jump in v(t) and these transients are longer if the model match is poor. Therefore it is advisable to update the model after a few iterations, collect new observation data, proceed with the iteration, etc.

Though there is no convergence proof for this approach either, simulation results indicate good convergence properties.

The Separation Procedure

The separation procedure in this case requires only the estimation of $u_w(t)$ from w(t). For a non-inferential controller this is

$$\hat{u}_w(t) = \frac{G_c(z)}{1+G_c(z)\hat{G}_p(z)} \quad w(t) \tag{19}$$

For the inferential controller,

$$\hat{u}_w(t) = \frac{G_F(z)}{\hat{G}_p(z)G_F(z)+G_M(z)[1-G_F(z)]} \quad w(t) \tag{20}$$

SIMULATION RESULTS

The two suggested methods have been thoroughly tested by computer simulation. In the following, some representative results will be shown. As a measure of accuracy, the quantity

$$q = \frac{||\underline{\hat{p}} - \underline{p}||}{||\underline{p}||} \tag{21}$$

has been evaluated for each run. The results obtained with ordinary least squares (i.e. in the first iteration step) will be shown for reference.

Instrumental Variable Method

The method was tested in closed loop with a second order process and a non-inferential controller. The process transfer function was

$$G_p(z) = \frac{0.208z^{-2} + 0.133z^{-3}}{1 - 1.036z^{-1} + 0.264\ z^{-2}}$$

while the controller was

$$G_c(z) = \frac{1 - 0.5z^{-1}}{1 - z^{-1}}$$

that is the discrete equivalent of the PI controller $G_c(s) = 1+0.25/s$

with $\Delta t = 2$ sec. The system was driven from the setpoint input with a unit amplitude PRBS and disturbed with

$$v(t) = \frac{1 - 0.679z^{-1} + 0.083z^{-2}}{1 - 1.036\ z^{-1} + 0.264\ z^{-2}}\ \omega(t)$$

where $\omega(t)$ is a $N(0,\sigma^2)$ white sequence. The sample consisted of 480 observations.

The results for different values of σ are shown in Table 1. The improvement over the least squares is about six to eight fold. The number of iteration steps needed was under 5.

Modified Least Squares

This method was tested in the inferential scheme. The process transfer function was

$$G_p(z) = \frac{0.139z^{-1} + 0.089z^{-2}}{1-1.036z^{-1} + 0.139z^{-2}}$$

A moving average filter was chosen with the transfer function

$$G_F(z) = 0.05z^{-1} + 0.25z^{-2} + 0.40z^{-3} + 0.30z^{-4}$$

The (initial) model transfer function was

$$G_M(z) = \frac{0.146z^{-1} + 0.134z^{-2}}{1 - 1.347z^{-1} + 0.423z^{-2}}$$

The system was driven from the setpoint input with a unit amplitude PRBS. A unit-step disturbance was applied to the output at a randomly selected moment during the first half of the observation period. The sample consisted of 500 observations.

Table 2 shows some results obtained with fixed model; several iterations were performed on the same set of observations. The improvement over least squares is about five-fold.

The data in Table 3 were obtained with updating the model in an outer iteration loop. Two iterations were performed with fixed model, then the model was updated, new observations collected (under noisy conditions), again two iterations performed with fixed model, etc. Under this scheme, the improvement over least squares is about ten-fold.

CONCLUSIONS

Two corrective methods have been developed for process identification in closed loop. Both are applicable to the inferential control scheme, under different assumptions. The algorithms are iterative but otherwise computationally simple. Simulation studies indicated fast convergence and a significant improvement in accuracy over the ordinary least-square technique. Further work is needed to obtain analytical proof and conditions for the convergence of the algorithms.

APPENDIX

Inferential Control [1]

The inferential control scheme is shown in Figure 3. Here $G_p(z)$ is the process transfer function

$$G_p(z) = \frac{B_p(z)}{1 + A_p(z)} = \frac{b_{p0}+b_{p1}z^{-1}+\ldots+b_{pn}z^{-n}}{1 + ap_1z^{-1}+\ldots+ap_nz^{-n}} \qquad \text{(A-1)}$$

$G_M(z)$ is the model transfer function

$$G_M(z) = \frac{B_M(z)}{1 + A_M(z)} = \frac{b_{M0}+b_{M1}z^{-1} + \ldots + b_{Mn}z^{-n}}{1 + a_{M1}z^{-1}+\ldots+ a_{Mn}z^{-n}} \qquad \text{(A-2)}$$

and $G_R(z)$ is the transfer function of the controller designed as

$$G_R(z) = \frac{G_F(z)}{G_M(z)} \qquad \text{(A-3)}$$

where

$$G_F(z) = \frac{B_F(z)}{1+A_F(z)} = \frac{b_{F1}z^{-1} + \ldots + b_{Fm}z^{-m}}{1 + a_{F1}z^{-1}+\ldots+a_{Fm}z^{-m}} \qquad \text{(A-4)}$$

the "filter" is the ideal closed-loop transfer function.

The closed-loop response of the system can be shown to be

$$y(t) = \frac{G_F(z)G_p(z)\ w(t) + [1-G_F(z)]\ G_M(z)\ v(t)}{G_F(z)\ G_p(z)+[1-G_F(z)]\ G_M(z)} \qquad \text{(A-5)}$$

If the filter is chosen so that its steady state gain $G_F(z)\Big|_{z=1} = 1$ then the steady state system response to step setpoint and noise is

$$\lim_{t\to\infty} y(t) = 1\ w_s(t) + 0v_s(t) \qquad \text{(A-6)}$$

That is, the controller is of integrating nature. On the other hand, if the model exactly matches the process, $G_M(z) = G_p(z)$, then the system response is

$$y(t) = G_F(z)\ w(t) + [1 - G_F(z)]\ v(t) \qquad \text{(A-7)}$$

In this sense, the inferential controller is of pole-placement type.

An equivalent non-inferential controller (Fig. 2) can be obtained as

$$G_c(z) = \frac{G_F(z)}{[1 - G_F(z)]G_M(z)} = \frac{B_F(z)[1 + A_M(z)]}{[1 + A_F(z)-B_F(z)]B_M(z)} \qquad \text{(A-8)}$$

It can be shown that $1 + A_F(z) - B_F(z)$ always contains the factor $1 - z^{-1}$ if $G_F(z)\Big|_{z=1} = 1$.

REFERENCES

[1] Brosilow, C. B. and M. Tong: Inferential Control of Processes - Part II: The Structure and Dynamics of Inferential Control Systems. AICHE Journal, 24, pp. 497-500 (1978).

[2] Gertler, J.: Direct Least Square Identification from Indirect Closed-Loop Measurements, Automatica. To be submitted.

[3] Gertler, J., C. B. Brosilow and C. Wu: Approaches to Closed-Loop Identification for Inferential Control. IFAC Congress, submitted (1984).

[4] Gustavsson, I., L. Ljung and T. Soderstrom: Identification of Processes in Closed Loop - Identifiability and Accuracy Aspects. Automatica 13, pp. 59-75 (1977).

[5] Soderstrom, T. and P. Stoica: Comparison of Some Instrumental Variable Methods - Consistency and Accuracy Aspects. Automatica, 17, pp. 101-115 (1981).

		a_1	a_2	b_2	b_3	q
true		-1.0360	0.2636	0.2081	0.1334	
$\sigma^2=0.5$	LS	-1.0548	0.2824	0.2084	0.1249	0.025
	IV	-1.0392	0.2668	0.2087	0.1319	0.004
$\sigma^2 = 1$	LS	-1.0873	0.3143	0.2092	0.1096	0.068
	IV	-1.0419	0.2696	0.2093	0.1308	0.008
$\sigma^2 = 2$	LS	-1.1322	0.3539	0.2161	0.0841	0.129
	IV	-1.0486	0.2744	0.2112	0.1297	0.016
$\sigma^2 = 5$	LS	-1.1699	0.3531	0.2677	0.0321	0.182
	IV	-1.0742	0.2889	0.2149	0.1303	0.035

TABLE 1

iteration	a_1	a_2	b_1	b_2	q
true	-1.0360	0.2640	0.1390	0.0890	
LS	-1.0968	0.3187	0.0973	0.0435	0.095
1	-1.0431	0.3131	0.1396	0.0898	0.046
2	-1.0314	0.2317	0.1382	0.0896	0.030
3	-1.0351	0.2376	0.1388	0.0901	0.024
4	-1.0344	0.2355	0.1388	0.0901	0.026
5	-1.0345	0.2358	0.1388	0.0901	0.026
6	-1.0345	0.2357	0.1388	0.0901	0.026

TABLE 2

model	iteration	a_1	a_2	b_1	b_2	q
	true	-1.0360	0.2640	0.1390	0.0890	
	LS	-1.0968	0.3187	0.0973	0.0435	0.095
1	1	-1.0431	0.3131	0.1396	0.0898	0.046
	2	-1.0314	0.2317	0.1382	0.0896	0.030
2	1	-1.0280	0.2656	0.1366	0.0883	0.008
	2	-1.0398	0.2810	0.1368	0.0872	0.016
3	1	-1.0406	0.2778	0.1370	0.0874	0.014
	2	-1.0388	0.2751	0.1370	0.0876	0.011
4	1	-1.0384	0.2752	0.1370	0.0876	0.011
	2	-1.0387	0.2757	0.1370	0.0876	0.011

TABLE 3

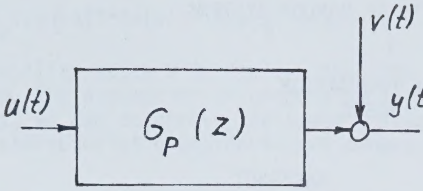


Fig. 1.

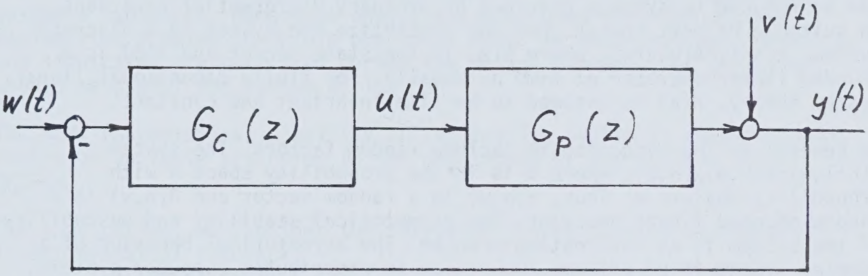

v(t)
w(t)
-
$G_C(z)$
u(t)
$G_P(z)$
y(t)

Fig. 2.

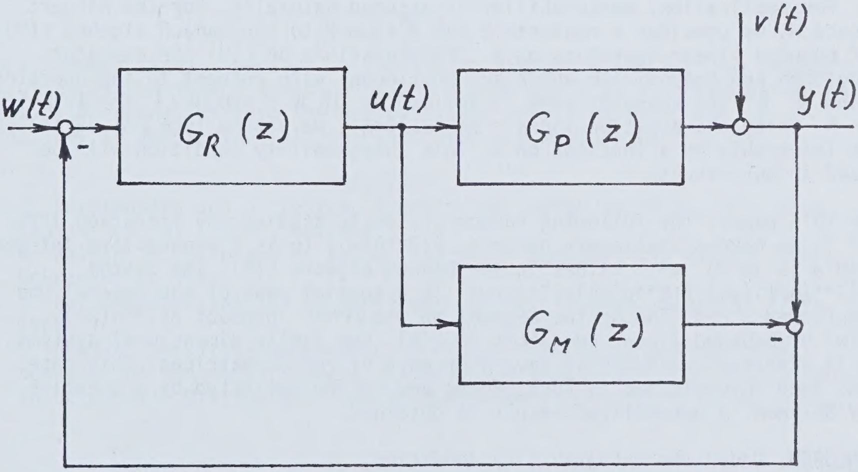

v(t)
w(t)
-
$G_R(z)$
u(t)
$G_P(z)$
y(t)
$G_M(z)$
-

Fig. 3.

ASYMPTOTICAL STABILITY OF RANDOM SYSTEMS

S. CHEN
NATIONAL SCIENCE FOUNDATION
WASHINGTON, D.C. 20550

ABSTRACT

We consider the asymptotical stability and unstability of the random linear system x(n+1,w)=A(n,w)x(n,w), where A(n,w) is a bounded linear operator (depending on time n and sample variable w) on a Hilbert space H of state vectors x(n,w).

SUMMARY

Linear systems governed by ordinary and partial differential equations can be reduced to systems governed by ordinary differential equations on suitable Hilbert spaces. One can discretize the system to a discrete system x(n+1)=A(n)x(n), where x(n) is the state vector and A(n) is a bounded linear operator at time n. Usually, for finite dimensional linear system theory, A(n) is assumed to be time-invariant and constant.

We generalize the situation to include random factors. The system x(n+1,w)=A(n,w)x(n,w), where w is in the probability space W with probability measure m. Thus, x(n,w) is a random vector and A(n,w) is a random bounded linear operator. The asymptotical stability and unstability of the system is an interesting problem. The asymptotical behavior of a complex system is of extreme importance to distributed computer systems, military control and guidance systems and economic systems. The analytic method provided in this paper should be supplemented by simulation and numerical methods. The combination of these three methods will be a powerful tool for system theory.

Let W be a probability space with probability measure m, that is m(W)=1. Let the time map T be a measurable map from W to W preserving the measure m. For application, measurability is assumed naturally. For the Hilbert space H, we consider a measurable map A from W to the Banach algebra L(H) of bounded linear operators on H. The operations on L(H) are operator addition and composition which are continuous with respect to the operator norm $\| \ \|$. The operator norm is defined by $\|B\| = \sup |Bx|$ for $\|x\|=1$ in H. Using the notation log+a = max(0,loga), we assume log+$\|A(.)\|$ to be integrable as a function on W. This integrability condition will be used in our result.

In this paper, the following random system is studied. By iteration T**n of T, we have a stationary process A(T**n(w)) (n is a nonnegative integer and w is in W) with values in the Banach algebra L(H). The system x(T**(n+1)(w))=A(T**n(w))x(T**n(w)) is a special case of the general one mentioned above. The system depends on the given product A(T**n(w))... A(w) of bounded linear operators in L(H). For finite dimensional systems, H is a vector space and we have a product of random matrices. This case has been investigated by Furstenberg and Kesten motivated by a question by Bellman. A generalized result is obtained.

THEOREM. Under the integrability condition,
$\chi(w,A,m) = \lim_{n\to\infty} (1/n)\log \|A(T^{**}n(w))\ldots A(Tw)A(w)\|$ is finite for almost all w in W.
If m is an ergodic measure, the limit is equal to a constant for almost all w in W which is the largest Lyapunov characteristic exponent

$$\chi(A,m)=\lim_{n\to\infty}(1/n)\int_W \log\| A(T^{**}n(w)\ldots A(w)\| \ m(dw).$$

REMARK. If the probability space W is finite, then the above integral becomes a finite sum. The assumption of ergodicity is not a severe restriction. In fact, we can decompose the space W into ergodic parts W_i, i=1,...,N, so that characteristic exponents are computed on ergodic parts respectively.

The system is defined to be turbulent or chaotic if $\chi(A,m)$ is positive. Intuitively, if $\chi(A,m)$ is positive, then the trajectories of the system are exponentially divergent. In other words, the system is asymptotically unstable. If $\chi(A,m)$ is negative, the system is asymptotically stable. These two possibilities are the two extreme cases. Between them, there exist a variety of intermediate cases. We can study various subspaces of H to have characteristic exponents defined on them to get stable and unstable subspaces. Numerical analysis can be used here for a given engineering problem to obtain practical data. In most problems, modeling and simulation are performed during the design process. Our criterion should be useful for system design.

THEOREM. (Asymptotical stability criterion) The system is asymptotically stable or unstable according to the negativity or the positivity of $\chi(A,m)$.

For application, we consider the following random control system with linear dynamical feedback

$$x(T^{**}(n+1)(w))=A(T^{**}n(w))x(T^{**}n(w))+B(T^{**}n(w))u(T^{**}n(w)),$$
$$u(T^{**}(n+1)(w))=C(T^{**}n(w))x(T^{**}n(w))+D(T^{**}n(w))u(T^{**}n(w)),$$

where x is the state vector and u is the control vector. The asymptotical stability or unstability of the system can be determined by the critrion. We may assume the control space U to be finite dimensional and the state space H to be a Hilbert space so that the direct sum $H\oplus U$ is a Hilbert space. A, B, C and D are required to be measurable functions on W such that

$$\begin{bmatrix} A(T^{**}n(w)) & B(T^{**}n(w)) \\ C(T^{**}n(w)) & D(T^{**}n(w)) \end{bmatrix}$$

is a bounded linear operator in the Banach algebra $L(H\oplus U)$. The feedback is given by the second equation of the system which is a dynamical system itself. We emphasize that the measurability should be valid in all practice. In particular, all constant maps are measurable.

REFERENCES

1. H. Furstenberg and H. Kesten, Products of random matrices, Ann. Math. Statistics, 31 (1960), 457-469.
2. R. Kalman, P. Falb and M. Arbib, Topics in Mathematical System Theory, McGraw-Hill, 1969.
3. S. Lefschetz, Stability of Nonlinear Control Systems, Academic Press, 1965.
4. D. Ruelle, Sensitive dependence of initial condition and turbulent behavior of dynamical systems, Bifurcation Theory and Its Applications in Scientific Disciplines, New York Academy of Sciences, 1979.

STOCHASTIC OPTIMAL CONTROL WITH FORCED SWITCHING ACTIONS

S. A. BELBAS
Electrical Engineering Department
University of Maryland
College Park, MD 20742

S. M. LENHART
Mathematics Department
University of Tennessee
Knoxville, TN 37996

ABSTRACT

We study the problem of optimal switching control of a stochastic system when there are state-dependent restrictions on the available switchings as well as forced switching actions. This problem leads to a system of nonlinear partial differential equations with unusual boundary conditions. Here we obtain a suitable penalization scheme and local regularity results.

1. INTRODUCTION AND STATEMENT OF THE PROBLEM

We consider a stochastic system that can be operated in any of a finite number of regimes. Let

$$A = \{1,2,\dots,m\}$$

be the set of regimes. The state space of the system is a bounded open set $\Omega \subset R^n$. With each regime $a \varepsilon A$ and each point $x \varepsilon \Omega$, there is associated a diffusion process $y^a_x(t)$, which is the state of the system with initial conditions given by the initial regime a and the initial state x. The process $y^a_x(t)$ is solution of the stochastic differential equation.

$$\begin{aligned} dy^a_x(t) &= m^a(y^a_x(t))dt + \\ &\quad + \sigma^a(y^a_x(t))dw(t); \qquad (1.1) \\ y^a_x(0) &= x \end{aligned}$$

The problem of optimal switching control of this system has been studied in [9]. For the problem considered in this paper, we impose the extra hard state constraints described below.

For every $a \varepsilon A$, we have a bounded open set $\Omega_a \subset \Omega$ such that:

(i) $\Omega = \Omega_a$ for some $a \varepsilon A$.

(ii) $\partial\Omega$ is smooth.

(iii) $\partial\Omega_a \setminus \partial\Omega$ is smooth.

The diffusion process $y^x_a(t)$ is defined for every $x \varepsilon \Omega_a$.

We shall describe below the requirements for an admissible policy

$$v_{a,x} = (\tau_1, a_1; \tau_2, a_2; \ldots)$$

where each τ_i is a random stopping time with respect to the family of σ-algebras $\{F_t^i\}$ generated by $\{w(t)\}$, and each a_i is a random variable, measurable with respect to F_{τ_i}, taking values in A.

(1). For $x \varepsilon \Omega$, let A_x denote the set of those indices a for which $x \varepsilon \Omega_a$. Then we require that a_i should take values in the set $A_{y_x^{a_{i-1}}}(\bar{\tau}_i)$.

(2). When $y_x^{a_{i-1}}(\bar{\tau}_x) \varepsilon \partial\Omega_{a_{i-1}} \setminus \partial\Omega$, where $\bar{\tau}_x$ is the time of first exit of the process from Ωa_{i-1}, then we require that $\tau_i = \bar{\tau}_x$ and $a_i \varepsilon A y_x^{a_{i-1}}(\bar{\tau}_x)$.

The heuristic interpretation of the above formal constraints is as follows: If $a \varepsilon A$ is the regime of the system and $y_x^a(t)$ is the state of the system, we only allow switchings such that the pair (state, regime) is always of the form (x,a) with $a \varepsilon A_x$.

The control actions satisfy the following requirements:

(1'). At each point $x \varepsilon \Omega$, the allowable switchings are those that take the system to some regime a such that $x \varepsilon \Omega_a$.

(2'). At the time of exit from Ω_a, we are forced to switch to some other regime, taking into account the restriction of (1) above.

The cost structure is the following:

There is a continuous rate of cost $f^a(x)$ associated with every regime $a \varepsilon A$, and a discrete switching cost k(a,b) for every time that a switching action is taken to change the regime of the system from a to b. Further, we assume for technical reasons that we have a discount factor $c^a(x) \geqslant 0$ for every regime $a \varepsilon A$. The total cost $J_{a,x}(v_{a,x})$ associated with a given switching control policy $v_{a,x}$ is:

$$\begin{aligned} J_{a,x}(v_{a,x}) = E_{ax} \Big\{ & \int_0^{\tau_x^\Omega} f^{a(t)}(y_x^{a(t)}(t)) . \\ & \cdot \exp\left(-\int_0^t c^{a(s)}(y^{a(s)}(s))ds\right)dt + \\ & + \sum_{i=1}^{N_{\tau_x}^{\Omega}} \left[k(a_{i-1}, a_i) \exp\left(-\int_{\tau_{i-1}}^{\tau_i} c^{a_{i-1}}(y_x^{a_{i-1}}(s)ds\right)\right]\Big\} \end{aligned} \tag{1.2}$$

where:

τ_x^Ω is the time of exit of the process y^a from Ω;

$N_{\tau_x^\Omega}$ is the (random) number of switchings;

$a(t) = a_i$ for $\tau_i \leqslant t < \tau_{i-1}$.

The minimal cost function associated with initial regime a is defined by

$$u^a(x) = \inf_{v_{a,x}} J_{a,x}(v_{a,x}) \tag{1.3}$$

where the infimum is taken over admissible control policies, satisfying the requirements (1), (2) above.

(Problems similar to the one considered here have been treated in [8,1]. These works dealt with forced stopping rather than forced switchings).

Let L^a be second-order elliptic operators of the form

$$L^a = - \sum_{i,j=1}^{n} a_{ij}^a(x) \frac{\partial^2}{\partial x_i \partial x_j} + \sum_{i=1}^{n} b_i^a(x) \frac{\partial}{\partial x_i} + c^a(x) I \tag{1.4}$$

where

$$b_i^a(x) = - m_i^a(x)$$

$$[a_{ij}^a(x)] = \frac{1}{2} [\sigma_{ik}^a(x) \sigma_{jk}^a(x)].$$

Then a formal application of the principle of dynamic programming [7] shows that $\{u^a_x\}$ must satisfy the following system of quasi-variational inequalities:

$$\begin{aligned} &\max \{L^a u^a(x) - f^a(x), u^a(x) - M^a u(x)\} = 0 \text{ for } x\varepsilon\Omega_a; \\ &u^a(x) = 0 \text{ for } x\varepsilon\partial\Omega \setminus \partial\Omega_a; \\ &u^a(x) = M^a u(x) \text{ for } x\varepsilon\partial\Omega_a \setminus \partial\Omega \end{aligned} \tag{1.5}$$

where

$$u(x) = \{u^a(x) : a \varepsilon A\};$$

$$M^a u(x) = \min_{\substack{b:b \neq a \\ b \varepsilon Ax}} \{u^b(x) + k(a,b)\} \tag{1.6}$$

2. PENALIZATION AND REGULARITY RESULTS

We use a penalization scheme related to the one employed in [9]. However, due to the peculiar character of the boundary conditions, we have to use additional terms that penalize the "forbidden" domains $\Omega \backslash \Omega a$. For earlier instances of "penalization of domains" see [4,12].

The natural approximation to (1.5) would be the following

$$L^a u^a_{\varepsilon\delta} + \sum_{\substack{b:b\neq a \\ b\varepsilon A_x}} \beta_\varepsilon (u^a_{\varepsilon\delta} - u^b_{\varepsilon\delta} - k(a,b)) =$$

$$= f^a(x) + \phi^a_\delta(x) \text{ in } \Omega; \tag{2.1}$$

$$u^a_{\varepsilon\delta} = 0 \text{ on } \partial\Omega$$

where β_ε is a smooth penalty function such that:

$$\beta'_\varepsilon \leqslant 0 \;, \; \beta''_\varepsilon \geqslant 0;$$

$$\beta_\varepsilon(t) \to +\infty \text{ as } \varepsilon \to 0 \text{ for } t > 0; \tag{2.2}$$

$$\beta_\varepsilon(t) = 0 \text{ for } t \leqslant 0;$$

$$|\beta_\varepsilon(t) - t\beta'_\varepsilon(t)| \leqslant C$$

and the functions ϕ^a_δ are smooth and have the properties:

$$\phi^a_\delta(x) = 0 \text{ , for } x\varepsilon\Omega_a \text{ ;}$$

$$\phi^a_\delta(x) \to +\infty \text{ , for } x\varepsilon\Omega\backslash\bar{\Omega}_a \text{ , as } \delta \to 0 \tag{2.3}$$

The approximation (2.1) has the drawback that the penalty terms are nonsmooth, since the sets A_x vary with x (although A_x is piecewise constant on connected components of the set $\Omega_a \setminus \left(\bigcup_{b:b\neq a}{}^{x} \partial\Omega_b \right)$.

Instead, we shall consider the following penalization scheme:

$$L^a u^a_{\varepsilon\delta} + \sum_{b:b\neq a} \beta_\varepsilon (u^a_{\varepsilon\delta} - u^b_{\varepsilon\delta} - k(a,b)) =$$

$$= f^a (x) + \phi^a_\delta(x) \quad \text{in} \quad \Omega; \tag{2.4}$$

$$u^a_{\varepsilon\delta} = 0 \quad \text{on} \quad \partial\Omega$$

It may not be obvious that this new penalization scheme actually takes into account constraint (1), and some explanations are in order.

We shall make the following assumption about the switching costs

$k(a,b)$: $\quad k(a,b) + k(b,c) > k(a,c)$

$$\text{for all } a,b,c\varepsilon\tilde{A} \text{ with } a\neq b \wedge b\neq c. \tag{2.5}$$

Condition (2.5) is stronger that the condition of no loops of zero cost [9]. Although the assumptions of [9] are sufficient for the case at hand, condition (2.5) will greatly simplify many of the arguments. We note that (2.5) implies that it is impossible to have two simultaneous switchings under the optimal policy determined by the solution of (1.5). Indeed, if $u^a(x) = u^b(x) + k(a,b)$ and $u^b(x) = u^c(x) + k(b,c)$, with both b and c being admissible regimes from a, then

$$u^a(x) \leqslant k(a,c) + u^c(x)$$

and therefore

$$k(a,c) \geqslant k(a,b) + k(b,c)$$

which contradicts (2.5).

As $\varepsilon \to 0$, we know from [9] that the solution $u^a_{\varepsilon\delta}$ of (2.4) converges in $C_1(\overline{\Omega})$ to u^a_δ solution of

$$\max\{L^a u^a_\delta - f^a - \phi^a_\delta , u^a_\delta - \tilde{M}^a u_\delta\} = 0 \quad \text{in} \quad \Omega;$$

$$u^a_\delta = 0 \text{ on } \partial\Omega \tag{2.6}$$

where $\tilde{M}^a$ is the extended implicit obstacle, viz.

$$\tilde{M}^a u(x) = \min_{b:b\neq a} \{u^b(x) + k(a,b)\} \tag{2.7}$$

i.e., constraint (1) is not taken into account in (2.7).

Now, let a_0 be such that

$$\Omega_{a_0} = \Omega.$$

Then, by a standard comparison argument we have

$$u_1^{a_0} \leqslant u_\delta^{a_0} \leqslant u_0^{a_0}, \quad \text{for } \delta \text{ sufficiently small,}$$

where

$\{u_1^a, a\varepsilon\tilde{A}\}$ is the solution of (2.7) for some arbitrary

but fixed value of δ, say $\delta = \delta_1 > 0$;

$u_0^{a_0}$ is the solution of

$$L^{a_0} u_0^{a_0} = f^{a_0} \quad \text{in } \Omega;$$

$$u_0^{a_0} = 0 \quad \text{on } \partial\Omega \tag{2.8}$$

Now it is easy to conclude that

$$u_1^a \leqslant u_\delta^a \leqslant k(a, a_0) + u_0^{a_0}$$

for δ sufficiently small, for all $a\varepsilon\tilde{A}$.

Thus, $||u_\delta^a||_{L^\infty(\bar{\Omega})} \leqslant C$, with a constant C independent of δ.

We shall show that it is impossible to have a sequence $\delta_n \to 0$ with

$$u_{\delta n}^a(x) = u_{\delta n}^b(x) + k(a,b)$$

for $x\varepsilon\bar{N}$, $\bar{N}$ being a ball in $\Omega_a \setminus \Omega_b$.

Indeed, if $x \in N$, and if

$$u^a_{\delta_n}(x) = u^b_{\delta_n}(x) + k(a,b),$$

then

$$u^b_{\delta_n}(x) < \tilde{M}^b u_{\delta_n}(x).$$

For δ_n sufficiently small, $\phi^b_{\delta_n} > K$, with K arbitrarily large. Consequently, on N $u^b_{\delta_n}$ satisfies

$$L^b u^b_{\delta_n} \geqslant f^a + K ;$$

$$u^b_1 \leqslant u^b_{\delta_n} \leqslant u_0^{a_0} + k(a_0, b) \text{ on } \partial N \qquad (2.9)$$

and thus

$$\max_N |u^b_{\delta_n}| \geqslant K'$$

with K' depending on N, K, $\|u^b_{\delta_n}\|_{L^\infty(\partial N)}$, and $K' \to \infty$ as $K \to \infty$.

This contradicts $\|u^b_\delta\|_{L^\infty(\Omega)} \leqslant C$.

In a similar way we can show that, for every

$$x \in (\Omega \setminus \tilde{\Omega}_a) \cap \left(\bigcup_{b : b \neq a} \Omega_b \right),$$

we must have

$$u^a_\delta = M^a u_\delta (x)$$

for δ sufficiently small.

These considerations suggest that the penalization scheme (2.4) takes into account both constraints (1) and (2). This claim will be justified rigorously once we have obtained regularity estimates for (2.4),

Let

$$\tilde{\Omega} = \Omega \setminus \left(\bigcup_{a \in \tilde{A}} \partial\Omega_a \right),$$

$$\tilde{\Omega}_a = \Omega_a \cap \tilde{\Omega} .$$

Local first derivative estimates for $u^a_{\varepsilon\delta}$ in $\tilde{\Omega}$ can be obtained as in [9].

We shall make first derivative estimates, assuming

$$\sup_{x\in B} |u^a_{\varepsilon\delta}(x)| \leq C_1 \quad , \quad \text{for } B \subset\subset \tilde{\Omega}_a .$$

We have the following:

Proposition 2.1. Let $x^0 \in \tilde{\Omega}_a$ and let $B(x^0)$ be a ball such that $B \subset\subset \tilde{\Omega}_a$. Then

$$\max_{b\in A_{x^0}} \|Du^b_{\varepsilon\delta}\|_{L^\infty(B)} \leq C = C(B).$$

Proof. Let be a smooth cutoff function such that

$$\zeta = 1 \text{ on } B$$

$$\zeta = 0 \text{ outside } B'$$

where $B \subset\subset B' \subset\subset \tilde{\Omega}_a$, B, B' being in the same connected component of $\tilde{\Omega}_a$. We define

$$w^b(x) = \zeta^2(x)\, u^b_{x_i}\, u^b_{x_i} + \lambda(u^b - C_1)^2$$

where λ will be selected appropriately.

Let b, x^1 be such that

$$w^b(x^1) = \max_c \max_\Omega w^c(x).$$

If $x^1 \notin B'$, the bound on w^b can be obtained from L^∞ - bound on u^b.

If $x^1 \in B'$, then, since $A_{x^1} = A_{x^0}$, we have by virtue of the maximum principle:

$$\begin{aligned}
0 &\leq -a^b_{ij}\, w^b_{x_i x_j} \leq -2\theta(\zeta\, u^b_{x_\ell})_{x_k}\, (\zeta\, u^b_{x_\ell})_{x_\ell} - \\
&- 2\lambda\theta\, u^b_{x_\ell}\, u^b_{x_\ell} + 2\lambda(u^b - C_1)(-a^b_{ij}\, (u^b)_{x_i x_j}) + \\
&+ 2\zeta\, u^b_{x_\ell}\, (-a^b_{ij}\, (\zeta u^b_{x_\ell})_{x_i x_j}) \leq \\
&\leq -2\theta(\zeta u^b_{x_\ell})_{x_k}\, (\zeta u^b_{x_\ell})_{x_k} - 2\lambda\theta u^b_{x_\ell}\, u^b_{x_\ell} + \\
&+ 2\lambda(u^b - C_1)[C\, Du^b + C - \sum_{a\in A_{x^0}} \beta(u^b - u^a - k(b,a))] + \\
&+ 2\zeta u_{x_\ell}\, [CD^2 u^b + CDu^b + C - \\
&- \sum_{a\in A_{x^0}} \beta'(\cdot)\, (u^b_{x_\ell} - u^a_{x_\ell})\, \zeta]
\end{aligned}$$

The terms involving β and β' are nonnegative:

$$\sum_{a\in A_{x^o}} \beta(u^b - u^a - k(b,a))\; 2\lambda\; (u^b - C_1)$$

$$+ \sum_{a\in A_{x^o}} \beta'(\cdot)\; \zeta\; (u^b_{x_\ell} - u^a_{x_\ell})(2\zeta\, u^b_{x_\ell}) \geqslant$$

$$\geqslant \sum_{a\in A_{x^o}} \beta'(\cdot)\; (w^b - w^a) - C$$

$$|\beta_\varepsilon (t) - \beta'_\varepsilon (t)\cdot t\; | \leqslant C.$$

Choosing λ sufficiently large, as in [9,10,11], we can conclude that

$$|Du^b(x^1)| \leqslant C.$$

Proposition 2.2. Under the assumptions of Proposition 1, we have

$$\max_{b\in A_{x^o}} ||D^2 u^b_{\varepsilon\delta}||_{L^\infty (B(x^o))} \leqslant C = C(B).$$

The proof follows by a slight modification of the method in [9, 10].

3. THE CASE OF COSTLESS SWITCHINGS

In case $k(a,b) = 0$ for all $a,b\in\tilde{A}$, the optimality condition corresponding to (1.5) is:

$$\max_{a\in A_x} \{L^a u(x) - f^a(x)\} = 0 \quad \text{for} \quad x\in\Omega_a;$$

$$u(x) \text{ continuous across } \partial\Omega_a\setminus\partial\Omega;$$

$$u(x) = 0 \text{ on } \partial\Omega \tag{3.1}$$

The penalized problem corresponding to (3.1) is:

$$L^a\, u^a_{\varepsilon\delta} + \sum_{b:b=a} \beta_\varepsilon\, (u^a_{\varepsilon\delta} - u^a_{\varepsilon\delta}) = f^a(x) + \phi^a_\delta\, (x) \quad \text{for} \quad x\in\Omega; \tag{3.2}$$

$$u^a_{\varepsilon\delta} = 0 \quad \text{on} \quad \partial\Omega$$

where β_ε, ϕ^a_δ are as in the previous section.

Let u_δ be the limit of $u^a_{\varepsilon\delta}$ in $C^1(\bar{\Omega})$, as $\varepsilon \to 0$. Then u^a_δ satisfies

$$\max_a \{L^a u_\delta - f^a - \phi^a_\delta\} = 0 \quad \text{for } x\varepsilon\Omega;$$

$$u_\delta = 0 \quad \text{on} \quad \partial\Omega \tag{3.3}$$

As in section 2, we can show local $W^{2,\infty}$ regularity of $u^a_{\varepsilon\delta}$ in the interior of each set over which A_x is constant, uniformly in ε, δ. Further, we have the following:

Let N be a ball contained in $\Omega\setminus\bar{\Omega}_a$. Then it is impossible to have $L^a u_{\delta_n} = f^a$ on N for any sequence $\delta_n \to 0$. This is shown as the corresponding assertion of section 2.

REFERENCES

[1]. Baras, J. S., "Constraints in stochastic unit commitment scheduling", presented at the U.S. Dept. of Energy, May 1982.

[2]. Belbas, S. A., "Optimal switching control of diffusion processes over a finite time horizon", Proc. 1982 Conf. Info. Sci. Systems, Princeton,NJ, pp. 164-169.

[3]. Bensoussan, A., and J. L. Lions, "Inequations quasi-variationnelles dependant d'un parametre", Annali Scuola Norm. Sup. Pisa, 16 (1977), 231-255.

[4]. Bensoussan, A., and J. L. Lions, "Applications des inequations variationnelles en controle stochastique", Dunod, Paris, 1978.

[5]. Evans, L. C., and A. Friedman, "Optimal stochastic switching and the Dirichlet problem for the Bellman equation", Trans. AMS 253, (1979), 365-389.

[6]. Evans, L. C., and P. L. Lions, "Resolution des equations de Hamilton-Jacobi-Bellman pour des operateurs uniformement elliptiques", C.R.A.S. Paris, 260, (1980), 1049-1052.

[7]. Fleming, W. H., and R. W. Rishel, "Deterministic and stochastic optimal control", Springer-Verlag, New York, 1975.

[8]. Kushner, H. J., "Probability methods for approximations in stochastic control and for elliptic equations", Academic Press, New York, 1977.

[9]. Lenhart, S. M., and S. A. Belbas, "A system of nonlinear partial differential equations arising in stochastic control with switching costs", SIAM J. on Applied Math.,43 (1983), 465-475.

[10].Lenhart, S. "Bellman Equation for Optimal Stopping Time Problems", Indiana Univ. Math J., 32, (1983), 363-375.

[11].Lions, P. L., These "Sur quelques classes d' equations aux derivees partielles non lineaires et leur resolution numerique", Paris, 1979.

[12].Lions, P. L., J. L. Menaldi, and A. S. Sznitman, "Construction de processus de diffusion reflechis par penalisation du domaine", C.R.A.S. Paris, Ser. I, 292, (1981), 559-562.

FREQUENCY DOMAIN CONDITIONS FOR THE GENERAL SERVOMECHANISM PROBLEM

D. OHM
AMAF Industries, Inc.
Columbia, Maryland 21045

and

S.P. BHATTACHARYYA AND J.W. HOWZE
Electrical Engineering Department
Texas A&M University
College Station, Texas 77843

ABSTRACT

This paper considers the problem of synthesizing controllers based on measurement feedback in linear multivariable systems so that a desired transfer function matrix $H_d(s)$ is obtained relating the disturbance input to the controller output. This problem is equivalent to the disturbance rejection problem for proper systems. A solution requires the existence of a stable proper $S(s)$ satisfying an equation of the form $\hat{G}(s)S(s)\hat{N}(s) = \hat{H}(s)-H_d(s)$ and the theory of valuations can be used to determine solvability.

1. INTRODUCTION

The General Servomechanism Problem is the problem of synthesizing controllers for linear multivariable systems so that a specified transfer function matrix $H_d(s)$ is obtained between external inputs (reference and disturbance) and the controlled output. This problem can be reformulated as a disturbance rejection problem.

The Disturbance Rejection Problem under state feedback was formalized and solved by Wonham [1] in the time domain setting. Extensions of this work to compensator synthesis under measurement feedback for strictly proper systems were obtained by Imai and Akashi [2], Willems and Commault [3], and Schumacher [4,5]. Equivalent frequency domain conditions for the state feedback problem were given by Bhattacharyya, Gomes and Howze [6]. The objective of this paper is to develop constructive frequency domain conditions for Disturbance Rejection in proper systems under measurement feedback. It is shown in [11] that a generalization to proper systems is essential in dealing with such problems as the servomechanism problem.

This research was supported by NSF Grant No. 8200852.

2. PROBLEM FORMULATION

Consider a system represented by the transfer function matrix equation

$$\begin{bmatrix} y_c(s) \\ y_m(s) \end{bmatrix} = \begin{bmatrix} G(s) & H(s)-H_d(s) \\ M(s) & N(s) \end{bmatrix} \begin{bmatrix} u_c(s) \\ u_r(s) \end{bmatrix} \tag{1}$$

where $u_c(s) \varepsilon R^m$ is the control input vector, $u_r(s) \varepsilon R^r$ is the disturbance input vector, $y_c(s) \varepsilon R^p$ is the output to be controlled and $y_m(s) \varepsilon R^q$ is measurement vector. We assume without loss of generality that the system is proper, but M(s) is strictly proper. Also we assume that the system is stabilizable from u_c and detectable from y_m. To this system, we add a compensator of the form

$$u_c(s) = C(s)\ y_m(s) \tag{2}$$

such that the closed-loop system transfer function

$$H_m(s) = G(s)\ [I - M(s)\ C(s)]^{-1}\ N(s) + H(s) - H_d(s)$$

is identically zero, and the closed-loop system is stable. This is clearly equivalent to realizing a desired $H_d(s)$ relating $u_r(s)$ to $y_c(s)$. Define a matrix

$$X(s) = C(s)\ [I - M(s)\ C(s)]^{-1}.$$

It is easy to show that X(s) is proper if and only if C(s) is proper. Thus the problem is to find a proper X(s) so that the closed-loop system is stable and $H_m(s) = 0$. Assume (1) is stabilizable from u_c and detectable from y_m; these properties are independent of stable $H_d(s)$.

3. SOLVABILITY CONDITIONS

Theorem 1. Given a system (1), disturbances can be rejected from the output y_c if and only if there exists a proper stable matrix X(s) such that

(1) $G(s)\ X(s)\ N(s) + [H(s) - H_d(s)] = 0$ (4)

and

(2) $P(s) = \begin{bmatrix} M(s) + M(s)\ X(s)\ M(s) & M(s)\ X(s) \\ X(s)\ M(s) & X(s) \end{bmatrix}$ is stable (5)

Corollary. For system (1), disturbances can be rejected from y_c if and only if the compensator satisifes

$$(1) \quad \begin{bmatrix} G(s) & H(s)-H_d(s) \\ M(s) & N(s) \end{bmatrix} = \begin{bmatrix} G(s) & 0 \\ M(s) & I \end{bmatrix} \begin{bmatrix} I & C(s) \\ M(s) & I \end{bmatrix}^{-1} \begin{bmatrix} I & 0 \\ M(s) & N(s) \end{bmatrix} \qquad (6)$$

(2) the matrix $\begin{bmatrix} I & C(s) \\ M(s) & I \end{bmatrix}^{-1}$ is stable.

Recent results by Vidyasagar et. al. [7] effectively characterize the class of all stabilizing, proper compensators for a given strictly proper system by using the ring of stable proper rational functions and defining right and left coprime factorizations

$$M(s) = N_M(s)\, D_M(s)^{-1} = \tilde{D}_M(s)^{-1}\, \tilde{N}_M(s) \qquad (7)$$

where $N_M(s)$, $D_M(s)$, $\tilde{N}_M(s)$ and $\tilde{D}_M(s)$ are all stable proper matrices. There exist stable proper $L(s)$, $R(s)$, $\tilde{L}(s)$, $\tilde{R}(s)$ such that

$$L(s)\, N_M(s) + R(s)\, D_M(s) = I$$

$$\tilde{N}_M(s)\, \tilde{L}(s) + \tilde{D}_M(s)\, \tilde{R}(s) = I.$$

By using these definitions all stabilizing compensators are characterized by [7],

$$C(s) = -[\tilde{L}(s) + D_M(s)\, S(s)]\, [\tilde{R}(s) - N_M(s)\, S(s)]^{-1}$$

where $S(s)$ is an arbitary stable proper matrix. It is straightforward to show that

$$X(s) = [D(s)\, S(s) + \tilde{L}(s)]\, \tilde{D}(s)$$

and we have the following theorem:

Theorem 2. Define

$$\hat{G}(s) = G(s)\, D_m(s), \quad \hat{N}(s) = \tilde{D}_m(s)\, N(s)$$

and

$$\hat{H}(s) = H(s) - G(s)\, \tilde{L}(s)\, \tilde{D}_m(s)\, N(s),$$

In the system (1), disturbances can be rejected from y_c if and only if there exists a stable proper solution S(s) such that

$$\hat{G}(s)\ S(s)\ \hat{N}(s) = \hat{H}(s) - H_d(s). \tag{8}$$

Our next result shows that the entire class of transfer functions $H_d(s)$ obtainable by connecting stabilizing proper controllers $\underline{C}$ is given as the class of $H_d(s)$ for which (8) with a fixed particular factorization (7) has a stable proper solution S(s).

Theorem 3.

The class of admissible transfer functions $H_d(s)$ for $\underline{S}$ is given by

$H_d = \{H_d(s) \mid \hat{G}(s)\ S(s)\ \hat{N}(s) = \hat{H}(s) - H_d(s)\}$ has a stable proper solution S(s)}.

3. EXISTENCE CRITERION FOR THE COMPENSATOR

Existence of a stable proper solution of the equation (8) can be checked by using valuation theory [8,9,10]. Define, for a scalar function g(s), $v_\alpha(g)$, the valuation of g at α, as the integer $v_\alpha(g)$ such that

$$g(s) = (s-\alpha)^{v_\alpha(g)}\ p(s)/q(s), \text{ where}$$

p(s), q(s) are not divisible by $s - \alpha$, and

$v_\infty(g)$: = deg. (denominator of g) - deg. (numerator of g).

For the matrix case, the i-th valuation of H(s) at α is defined by

$$v_\alpha^{(i)}(H) = \min_i\ v_\alpha(H^{(i)})$$

where $H^{(i)}$ denotes an ixi minor of H(s). It is a well-known result [10] that there exists a solution Y(s) which has no poles at α, for the following equation, with A(s) of full column rank r,

$$Y(s)\ A(s) = B(s)$$

if and only if

$$v_\alpha^{(r)}\ [A(s)] = v_\alpha^{(r)} \begin{bmatrix} B(s) \\ A(s) \end{bmatrix}.$$

By using the Kronecker product, the existence of a stable, proper solution of (8) can be checked as

$$v_\alpha^{(n)}\left[\hat{G}^T(s) \otimes \hat{N}(s)\right] = v_\alpha^{(n)}\begin{bmatrix}\hat{G}^T(s) \otimes \hat{N}(s)\\ \hat{H}^T(s) - H_d(s)\end{bmatrix}, \ \mathrm{Re}\ \alpha \geq 0,\ \alpha = \infty$$

where n is the rank of $\hat{G}^T(s) \otimes \hat{N}(s)$.

4. EXAMPLE

Given the system in Figure 1 characterize all admissible $H_d(s)$.

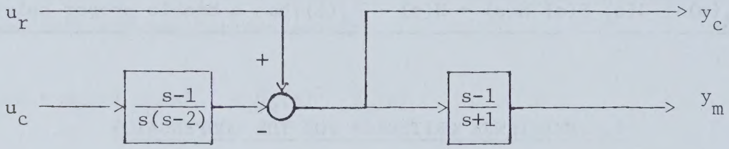


Fig. 1

The system matrices are

$$\begin{bmatrix} G(s) & H(s) \\ M(s) & N(s) \end{bmatrix} = \begin{bmatrix} \frac{-(s-1)}{s(s-2)} & 1 \\ \frac{-(s-1)^2}{s(s+1)(s-2)} & \frac{s-1}{s+1} \end{bmatrix}.$$

Note that the system is stabilizable from u_c and detectable from y_m. Now M(s) can be factored (7) as

$$M(s) = \begin{bmatrix} \frac{s(s-2)}{(s+1)^2} & -1 \end{bmatrix} \begin{bmatrix} \frac{-(s-1)^2}{(s+1)^3} \end{bmatrix}, \text{ and}$$

$N_m(s) = \tilde{N}_m(s) = \frac{-(s-1)^2}{(s+1)^3}$, $D_m(s) = \tilde{D}_m(s) = \frac{s(s-2)}{(s+1)^2}$, $L(s) = \tilde{L}(s) = \frac{-(40s+1)}{s+1}$ and $R(s) = \tilde{R}(s) = \frac{s^2-34s\ 17}{(s+1)^2}$. Now $\hat{G}(s) = \frac{-(s-1)}{(s+1)^2}$, $\hat{N}(s) = \frac{s(s-1)(s-2)}{(s+1)^3}$, $\hat{H}(s) = \frac{(s+1)^4-(s-1)^2(40s+1)}{(s+1)^4}$, and $\hat{G}^T(s) \otimes \hat{N}(s) = \frac{-s(s-1)^2(s-2)}{(s+1)^5}$.

Valuations of $G^T(s) \otimes N(s)$ are $v_\alpha = \begin{cases} 1\ , & \alpha = \infty \\ 1\ , & \alpha = 0 \\ 2\ , & \alpha = 1 \\ 1\ , & \alpha = 2 \\ 0\ , & \text{otherwise in the RHP.} \end{cases}$

If $H_d(s) = \frac{n(s)}{d(s)}$, then

$$\hat{H}(s) - H_d(s) = \frac{(s+1)^4[d(s) - N(s)] - (s-1)^2(40s+1)\ d(s)}{(s+1)^4\ d(s)} = \frac{\hat{n}(s)}{\hat{d}(s)}\ .$$

Now $v_\infty \begin{bmatrix} \hat{G}^T(s) \otimes \hat{N}(s) \\ \\ \hat{H}(s) \end{bmatrix}$ must be unity, so clearly

$\deg \hat{d}(s) - \deg \hat{n}(s) \geq 1$, or equivalently $\deg n(s) = \deg d(s)$ is required and the leading coefficients of $n(s)$ and $d(s)$ must be equal. The v_∞ constraint guarantees casuality of $C(s)$). Next, constraints $\hat{n}(0) = 0$, $\hat{n}(1) = 0$, $\frac{d}{ds}(\hat{n}(s)) \Big|_{s=1} = 0$ and $\hat{n}(2) = 0$ follow from valuations v_0, v_1 and v_2. If $n(s) = s^q + \alpha_{q-1} + \cdots + \alpha_0$ and $d(s) = s^q + \beta_{q-1} s^{q-1} + \cdots + \beta_0$, then these constraints on $\hat{n}(s)$ require the following conditions:

i) $\alpha_0 = 0$ (tracks steps),

ii) $\beta_{q-1} + \cdots + \beta_0 = \alpha_{q-1} + \cdots + \alpha_0$,

iii) $(q-1)(\beta_{q-1} - \alpha_{q-1}) + (q-2)(\beta_{q-2} - \alpha_{q-2}) + \cdots + (\beta_1 - \alpha_1) = 0$,

iv) $2^{q-1}\alpha_{q-1} + 2^{q-2}\alpha_{q-2} + \cdots + 2\alpha_1 + \alpha_0 = -2^q$

($s-2$ must divide $n(s)$) and v) $d(s)$ must be Hurwitz. It is clear from this example that even in the scalar case, arbitrary choices of $H_d(s)$ are almost never admissible. Further constraints may be imposed on the coefficients of $H_d(s)$ depending on desired response characteristics, and in general a constrained parameter optimization scheme could be employed.

Suppose it is required that the system track ramp inputs u_r. Then in addition to i) - v), it is required that vi) $\alpha_1 = 0$. Based on constraints i) - vi), it can be shown that $q \geq 4$ is required; one admissible $H_d(s)$ is given by $H_d(s) = \frac{s^2(s-2)(s-445)}{(s+2)^2(s+3)(s+\frac{34}{3})}$. Then

$$X(s) = \frac{4s(s-2)(s+1)(349s+102)}{(s+2)^2(s+3)(s+\frac{34}{3})}$$, and the required compensator is

$$C(s) = \frac{4(s+1)(349s+102)}{3s(s-445)} .$$

5. CONCLUDING REMARKS

It is found that the General Servomechanism Problem is equivalent to the measurement feedback disturbance rejection problem, and the solution is closely related to the existence of a stable, proper solution of the rational matrix equation

$$A(s)\ X(s)\ B(s) = C(s).$$

This equation is crucial in linear control systems using output feedback. Existence criteria for stable, proper solutions of this equation can be established by using valuation theory. The conditions for the existence of stable proper $\hat{X}(s)$ satisfying (8) are the frequency domain counterparts of generalized stabilizing (C,A,B) pairs that arise in the stage space version of this problem [11].

6. REFERENCES

[1] W. Wonham, "Linear Multivariable Control - A Geometric Approach", 2nd Ed., New York; Springer Verlag, 1979.

[2] H. Imai and H. Akashi, "Disturbance Localization and Pole Shifting by Dynamic Compensation", IEEE Trans. Auto., Contr., Vol. AC-26, pp. 226-235, Feb. 1981.

[3] J.C. Willems and C. Commault, "Disturbance Decoupling by Measurement Feedback with Stability or Pole Placement", SIAM J. Contr. Optim., Vol. 19, No. 4, pp. 490-504, July 1981.

[4] J.M. Schumacher, "Compensator Synthesis using (C,A,B) pairs", IEEE Trans. Auto. Contr., Vol. AC-25, No. 6, pp. 1133-1138, Dec. 1980.

[5] J.M. Schumacher, "Regulator Synthesis using (C,A,B) pairs", IEEE Trans. Auto. Contr., Vol. AC-27, No. 6, pp. 1211-1221, Dec. 1982.

[6] S.P. Bhattacharyya, A.C. del nero Gomes and J.W. Howze, "The Structure of Robust Disturbance Rejection Control", IEEE Trans. Auto. Contr., Vol. AC-28, No. 5, to appear.

[7] M. Vidyasagar and N. Viswanadham, "Algebraic Design Techniques for Reliable Stabilization", IEEE Trans. Auto., Contr., Vol. AC-27, No. 5, pp. 1085-1095, Oct. 1982.

[8] G.D. Forney, "Minimal bases of Rational Vector Spaces, with Applications to Multivariable Linear Systems", SIAM J. Contr., Vol. 13, pp. 493-520, 1975.

[9] S. Kung and T. Kailath, "Some Notes on Valuation Theory in Linear Systems", Proceedings of IEEE Conference on Decision and Contr., San Diego, CA, Jan. 1979.

[10] T. Kailath, "Linear Systems", Prentice Hall Inc., Englewood Cliffs, 1980.

[11] D. Ohm, J.W. Howze, S.P. Bhattacharyya, "Generalized (C,A,B) Paris and the Linear Servomechanism Problem", to be presented at the 1983 American Control Conference, June 1983, San Francisco, CA.

ON THE TRACKING CONTROLLER DESIGN

J. KIM AND D.H. CHYUNG
Electrical and Computer Engineering
University of Iowa
Iowa City, Iowa 52242

ABSTRACT

The problem of synthesizing a controller which tracks the given reference input with zero steady state error is investigated. The system is assumed to be given by a transfer function.

Suppose a linear control system is given by the transfer function

$$G(s) = \frac{q(s)}{p(s)} = \frac{b_m s^m + b_{m-1} s^{m-1} + \ldots b_1 s + b_o}{s^n + a_{n-1} s^{n-1} + \ldots + a_1 s + a_o}, \quad m < n$$

and let U(s) be the input and Y(s) be the output.

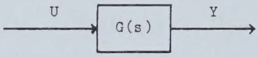
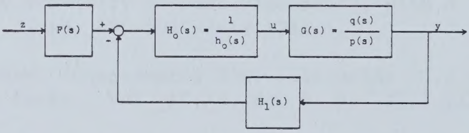


Fig. 1 Linear Control System

The problem is to synthesize a control u(t) so that the resulting system response y(t) follows a given reference function z(t) with zero steady state error, that is, y(t) = z(t) in steady state.

Assume that all state variables are measurable, that is, $s^j Y(s)$ is available for feedback for all j, $0 \leq j < n$. If some of the state variables are not directly observable, then they may be reconstructed from the observable variables by the method given in reference 1.

It is shown that the controller given in Fig. 2 satisfies the tracking requirement.



$$H_1(s) = k_{n-1}s^{n-1} + k_{n-2}s^{n-2} + \ldots + k_1 s + k_0$$

$$H_o(s) = \frac{1}{h_o(s)} = \frac{1}{k_{n+m}s^m + k_{n+m-1}s^{m-1} + \ldots + k_{n+1}s + k_n}$$

$$F(s) = c_n s^n + c_{n-1}s^{n-1} + \ldots c_1 s + c_0$$

Fig. 2 Tracking Controller

The constants $k_o, \ldots, k_{n+m}$ are chosen such that the closed loop system has poles at the desired locations. Of course, the complex poles must be assigned as complex conjugate pairs. It can be proved that these arbitrary pole assignments are always possible. Once the constants, and hence $H_o(s)$ and $H_1(s)$, are determined, then the constants $c_o, c_1, \ldots, c_n$ are chosen such that $F(s)q(s) = h_o(s)p(s) + q(s)H_1(s)$.

Furthermore, if the r-th derivative of the reference input z(t) is zero, that is $z^{(r)}(t) = 0$, then c_i may be replaced by zero for all $i \geq r$.

EXAMPLE. As an example, consider the system $G(s) = \dfrac{(s+1)}{s^2(s+2)}$. It is required that the system track a ramp reference input with zero steady state error. Since n=3, m=1 and r=2,

$$H_o(s) = \frac{1}{s + k_3}\ , \quad H_1(s) = k_2 s^2 + k_1 s + k_o\ , \quad F(s) = c_1 s + c_o$$

and the order of the closed loop system is 4. Choose the controller parameters such that the closed-loop characteristics are at -2, -0.45, $-0.65 \pm j0.8$. Then the resulting control system is as shown in Fig. 3. Fig. 4 shows the system response to a ramp reference input, and Fig. 5 shows the output response to a step reference input.

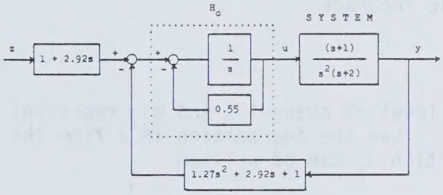
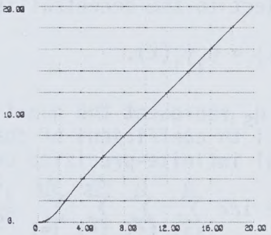
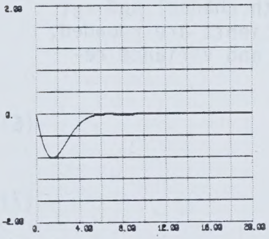


Fig. 3 Tracking controller



(a) Ramp input response

Fig. 4 Ramp input response



(b) Error

Fig. 4 (continued)

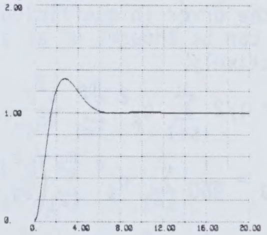


Fig. 5 Step response

REFERENCES

1. D.H. Chyung, On a Method for Reconstructing Inaccessible State Variables Using Time Delays, 8th IFAC World Congress, August, 1981, Kyoto, Japan.
2. J.B. Pearson, Compensator Design for Dynamic Optimization, Int. J. on Control, Vol. 9, No. 4, 1969.

THE COEFFICIENT WORDLENGTH PROBLEM IN DECENTRALIZED/HIERARCHICAL MULTIPROCESSOR CONTROL SYSTEMS

M. BURNS, Ü. ÖZGÜNER
The Ohio State University
Department of Electrical Engineering
Columbus, OH 43210

ABSTRACT

This paper examines the effects of wordlength selection for decentralized/hierarchical multiprocessor systems for LQG control.

Consider the linear time invariant system with N states and ν channels, each channel having r_i inputs and K_i outputs, i = 1, 2, 3, ... ν.

$$x(k+1) = \Phi x(k) + \sum_{i=1}^{\nu} \Gamma_i u_i(k) + w_i(k) \tag{1}$$

$$y_i(k) = L_i x(k) + w_{2i}(k) \tag{2}$$

Let the control strategy be to minimize the quadratic cost function

$$J = \lim_{m\to\infty} E\left\{\frac{1}{m}\sum_{k=1}^{\nu}[x^T(k)Qx(k) + \sum_{i=1}^{\nu} u_i^T(k)R_i u_i(k)]\right\} \tag{3}$$

subject to the decentralized static feedback

$$u_i(k) = G_i y_i(k). \tag{4}$$

Let q_s represent the quantization level of channel s and g_{si} represent the i th coefficient of channel s. Then the degradation in J from the ideal infinite-precision cost function J_∞ can be written

$$dJ = 1/2 \sum_{s=1}^{\mu}\sum_{t=1}^{\mu}\sum_{i=1}^{m_s}\sum_{j=1}^{m_t} \frac{\partial^2 J_\infty}{\partial g_{si}\partial g_{tj}}\,\varepsilon_{si}\,\varepsilon_{tj} \tag{5}$$

where m_s equals the number of coefficients of the s th channel and ε_{si} is the error in coefficient g_{si}. Because the coefficients are rounded, ε_{si} can be thought of as a random variable with mean and variance respectively

$$\overline{dJ} = 1/2 \sum_{s=1}^{\nu} q_s^2 \sum_{i=1}^{m_s} \frac{\partial^2 J_\infty}{\partial g_{si}^2} \tag{6}$$

$$\sigma^2_{dJ} = \frac{1}{960}\sum_{s=1}^{\nu} q_s^4 \sum_{i=1}^{m_s}\left(\frac{\partial^2 J_\infty}{\partial g_{si}^2}\right)^2 + \frac{1}{288}\sum_{s=1}^{\nu-1}\sum_{t=s+1}^{\nu} q_s^2 q_t^2 \sum_{i=1}^{m_s}\sum_{j=1}^{m_t}\left(\frac{\partial^2 J_\infty}{\partial g_{si}\partial g_{tj}}\right)^2 . \tag{7}$$

If it is desired that the degradation in J be less than some specified amount E_o, one can achieve a 95% chance of this by choosing q_i such that

$$\overline{dJ} + 2\sigma_{dj} \leq E_o \tag{8}$$

so equation (6-8) specify a test to determine if a given choice of processor wordlengths.

References and further details on wordlength for decentralized dynamic and delayed-sharing systems may be obtained from the authors. The research for this paper has been supported in part by grant #ECS-8203877 from The National Science Foundation.

TIME DOMAIN DESIGN OF ROBUST CONTROLLERS FOR LINEAR REGULATORS

Rama Krishna Yedavalli
Asst. Prof., Dept. of Mechanical Engineering
Stevens Institute of Technology
Hoboken, N.J. 07030

Introduction

The main objective of the paper is to present a design algorithm that incorporates both "Stability Robustness" and "Performance Robustness" aspects into the time domain design of controllers for linear regulators. The algorithm takes the perspective that in a realistic design a trade-off between stability and performance is to expected. First, a stability robustness condition in time domain is presented and it is shown that the proposed condition is less conservative than the corresponding frequency domain condition. Then simple measures of 'stability robustness' and 'performance robustness' are developed and an algorithm that achieves the said trade off is proposed.

Time Domain Stability Robustness Analysis

Let us consider, for simplicity, the linear time invariant regulator

$$\dot{x} = Ax + Bu \quad x(0) = x_o,\ x\varepsilon R^n,\ u\varepsilon R^m$$

with a state feedback controll u = Gx and evaluated by the standard quadratic index

$$J_N = \int_o^\infty (x^T Q\, x + \rho_c u^T u)dt$$

The closed loop system is

$\dot{x} = \bar{A}\, x$ and is asymptotically stable, $\bar{A} = A + BG$.

Let there be an additive perturbation E so that the perturbed system matrix is $\bar{A} + E$. In [2] (which is an updated version of [1] with corrections and typos taken care of), it is shown that

Theorem A: The perturbed system matrix $\bar{A} + E$ is stable for all perturbations E_{ij} satisfying

$|E_{ij}| \leq \Delta_{ij}$ where Δ_{ij} is the magnitude of the maximum deviation expected, if

$$\rho[(E_m A_m^{-1})_s] < 1$$

where $\bar{A} = A_d + A_e$, $A_d = \text{Diag}[\text{Re}\,\lambda_i(\bar{A})]$, $E_m = (|A_e| + \Delta)_s$

$A_m \doteq |A_d|$ and ρ is the spectral radius and $[\cdot]_s$ denotes the symmetric part of matrix $[\cdot]$ and $|[\cdot]|$ denotes the modulus matrix and λ_i=eigenvalues.

Note that if ΔA and ΔB are the matrices with magnitudes of maximum expected perturbations in A and B respectively then the magnitude of the maximum deviation in $\bar{A} = A + BG$ is taken as $\Delta = (\Delta A + \Delta B|G|)$.

The following example shows that the proposed time domain condition is less conservative than the corresponding frequency domain condition [3] for stability of a matrix $A + \Delta A$, namely

$$\sigma_{min}[j\omega I - A] > \sigma_{max}[\Delta A] \text{ for all } \omega > 0$$

$\sigma[\cdot]$ = singular value of $[\cdot]$.

Example: $A = \begin{bmatrix} -8 & 0 \\ 0 & -1 \end{bmatrix}$; $\Delta A = \begin{bmatrix} 0 & 1.5 \\ 1.5 & 0 \end{bmatrix}$

At $\omega = 1$ $\quad \sigma_{min}[j\omega I-A] = 1.414 \ngtr \sigma_{max}[\Delta A] = 1.5$

But with $A_d = A$ and $E_m = \Delta A$

$$\rho[(E_m A_m^{-1})_s] = 0.84375 < 1 \rightarrow A + \Delta A \text{ is stable.}$$

Thus the proposed time domain condition succeeds in predicting the stability of $(A + \Delta A)$ whereas the frequency domain condition fails to do so.

Robust Controller Design. It is clear that a variety of controller gains may satisfy the stability condition for given perturbations. In order to compare different controllers (or models) from stability robustness point of view, we now define a measure of stability robustness called "Stability Robustness Index, $\beta_{S.R.}$"

$$\beta_{S.R.} \overset{\Delta}{=} ||Re\lambda_{min}(\bar{A})| - |Re\lambda_{min}(\bar{A}_p)|| \,/\, |Re\,\lambda_{min}(\bar{A})|$$

where $\bar{A}_p = \bar{A} + (\Delta A + \Delta BG)$ and it is assumed that the control gain G is such that the condition of theorem A is satisfied (which in turn implies $\bar{A}_p$ is stable). Thus $\beta_{S.R.} = 0$ corresponds to a highly robust system from stability point of view. Similarly, we also define a measure of performance robustness, namely

$$\beta_{P.R.} \overset{\Delta}{=} |J_p - J_N| \,/\, J_N$$

where J_p is the performance index for the perturbed system (with $\bar{A}_p$ as the plant matrix). Thus $\beta_{P.R.} = 0$ corresponds to a highly robust system from performance point of view. The control design algorithm involves the use of ρ_c as a design variable and calculate $\beta_{S.R.}$ and $\beta_{P.R.}$ for different controllers. By plotting the curves of i) $\beta_{S.R.}$ vs. $\beta_{P.R.}$ and ii) the nominal regulation costs $(x^Tx)^{\frac{1}{2}}$ vs. $(u^Tu)^{\frac{1}{2}}$, one can pick a controller that achieves a satisfactory trade-off.

Example: Scalar system $\dot{x} = -x + u$, $x_o = 1$, $J_N = \int_o^\infty (x^2 + \rho_c u^2)dt$

Let $|\Delta a| = 0.5$, $|\Delta b| = 1.207$.

The results are given in Table 1 (Note: all the controllers corresponding to range of ρ_c considered satisfy the stability robustness condition).

Table 1

ρ_c	1/8	1/4	1/2	3/4	1	5/4	3/2	2
$\beta_{S.R.}$	0.63	0.44	0.22	0.09	0	0.06	0.12	0.19
$\beta_{P.R.}$	0.39	0.311	0.18	0.08	0	0.07	0.13	0.23
$(x^Tx)^{\frac{1}{2}}$	0.418	0.473	0.54	0.57	0.59	0.61	0.62	0.64
$(u^Tu)^{\frac{1}{2}}$	0.823	0.58	0.39	0.30	0.25	0.21	0.18	0.14

A satisfactory robust controller could be one corresponding to $\rho_c = \frac{1}{2}$ or 3/4 or 1.

Acknowledgement

This research was sponsored by AFOSR/AFSC of USAF under grant #AFOSR-83-0139.

References

1. Yedavalli, R.K., Banda, S.S. and Ridgely, D.B., "Time Domain Analysis for Stability Robustness of Large Scale LQG Regulators", Proceedings of the 1983 AIAA Guidance & Control Conf., Gatlinburg, TN, Aug. 15-18, 1983, pp. 815-822.

2. Ibid: Under Review for possible publication in the AIAA Journal of Guidance, Control and Dynamics.

3. Lee, W.H., Gully, S.W., Eterno, J.S. and Sandell, N.R., Jr., "Structural Information in Robustness Analysis", Proceedings of American Control Conference, 1982, pp. 1040-1045.

MINIMUM NORM POLE ASSIGNMENT VIA SYLVESTER'S EQUATION

L.H. KEEL, J.A. FLEMING, and S.P. BHATTACHARYYA
Department of Electrical Engineering, Texas A&M University
College Station, Texas 77843

ABSTRACT

The Sylvester equation parameterization of the multi-input pole assignment problem is used in an algorithm to minimize the norm of the state feedback matrix. This algorithm is based on stable and efficient numerical procedures.

DEVELOPMENT OF THE ALGORITHM

In [1] a new procedure for pole assignment was presented. It consists of solving for X, and then for F, the equations

$$AX - X\widetilde{A} = -BG \quad \text{and} \quad FX = G \tag{1}$$

for given $(A,B,\widetilde{A})$ with an arbitrary choice of $\underline{G}$. From [2], X generically has full rank if (A,B) is controllable and $(G,\widetilde{A})$ is observable. It is seen that F has the property

$$\sigma(A + BF) = \sigma(\widetilde{A}). \tag{2}$$

The advantages of this procedure are: (a) the algebraic variety $F(\Lambda)$ of matrices F which assign a prescribed set of eigenvalues Λ can be obtained by setting $\Lambda=\sigma(\widetilde{A})$ for a fixed $\widetilde{A}$, letting the free parameter G run through the set of all possible real values, and (b) efficient numerical procedures [3] are available for the solution of equation (1).

Based on the preceding, an algorithm was given in [4] for optimizing the conditioning of the closed loop eigenvectors. That work was motivated by a related but different parameterization of $F(\Lambda)$ by Moore [5]. In [6] eigenvalues are placed sequentially with the feedback matrix minimized at each stage, and in [7] the single input case has been treated.

In this paper we summarize an algorithm which uses the Sylvester equation parameterization of the pole assignment problem [1] in order to minimize the Frobenius norm of F. Since pole assignment is an important step in most state space based synthesis problems [10], the results given here may have wide application.

Let (A,B) be a controllable pair and let Λ be a set of n self-conjugate complex numbers. Consider the problem of choosing F so that

$$\sigma(A + BF) = \Lambda \tag{3}$$

with F minimizing the performance index

$$I = \sum_{i=1}^{r} \sum_{j=1}^{n} f_{ij}^2 = \text{trace}\,(F^t F). \tag{4}$$

The gradient of I with respect to G (the parameterization matrix of (1)) can be explicitly calculated, and used in an algorithm for minimum norm pole placement as follows:

Step 0. Set k = 1, pick $G_1 \in R^{rxn}$

Step 1. Solve for X_k: $\quad A X_k - X_k \widetilde{A} = -BG_k$. $\quad$ (5)

Step 2. Obtain F_k and H_k: $\quad F_k X_k = G_k$ and $H_k X_k^t = F_k$. $\quad$ (6)

Step 3. Compute U_k: $\quad \widetilde{A} U_k - U_k A = H_k^t F_k$. $\quad$ (7)

Step 4. Compute $\dfrac{\partial I}{\partial G} = 2\,(H_k - B^t U_k^t)$. $\quad$ (8)

If $||\partial I/\partial G||$ is greater than a prespecified value ε go on to Step 5; ELSE take $F = F_k$. STOP.

Step 5. Obtain a new value G_{k+1} using a gradient based method. Set $k = k + 1$ and return to Step 1.

NUMERICAL CONSIDERATIONS

The computations required for steps 1 and 3 of our algorithm are performed using the Hessenburg-Schur method as described in [3]. Note: (a) The matrix X is determined uniquely if and only if the eigenvalues of A and $\tilde{A}$ are disjoint; if, in addition, (A,B) is a controllable pair and $\tilde{A}$ is cyclic [2] then X is nonsingular for almost all G. (b) If $\tilde{A}$ is chosen to be quasi-diagonal (real poles on the diagonal, complex conjugate poles formed by 2x2 real blocks) the method of [3] may be simplified, speeding up computation by a factor of two. (c) An error analysis [3] for the Hessenberg-Schur method has shown this procedure to be numerically stable.

Step 2 of the algorithm uses the singular value decomposition. This is known to be a stable procedure [8], and the condition number of X is an easily computed byproduct.

Step 5 uses a quasi-Newton method for unconstrained minimization as described in [9]. Based on the gradient information and an updated estimate of the Hessian matrix, the procedure chooses a descent direction and step size so that the functional is decreased.

The algorithm has been programmed in FORTRAN and run on a VAX 11/780 system. Several examples have demonstrated that for multi-input systems direct pole placement by the Sylvester Equation method (without optimization) is neither inherently better nor worse than the traditional pole placement method as in Wonham [10]. However, substantial improvement may be made by *minimizing* over the G matrices that parameterize the pole placement process through the use of the algorithm described here.

Future research includes: (a) minimum $||F||$ pole placement for eigenvalues specified to lie in allowable REGIONS of the complex plane. In our formulation this would allow freedom in the matrices $\tilde{A}$ and G. (b) Application of the Sylvester parameterization in the direct design of multivariable compensators incorporating robustness, tracking, and other design considerations.

REFERENCES

[1] Bhattacharyya, S.P., and E. deSouza, "Pole Assignment via Sylvester's Equation", *Syst. and Contr. Letters*, Vol 1(4):261-263, Jan. 1982.

[2] deSouza, E., and S.P. Bhattacharyya, "Controllability, Observability, and the Solution of AX-XB=C", *Lin. Alg. and Its Applic.*, 39:167-188, 1981

[3] Golub, G.H., S. Nash, and C. VanLoan, "A Hessenberg-Schur Method for the Problem AX+XB=C", *IEEE Trans. Aut. Cont.*, AC-24(6):909-913,Dec1979

[4] Cavin, R.K.III, and S.P. Bhattacharyya, "Robust and Well Conditioned Eigenstructure Assignment via Sylvester's Equation", *Proc. American Control Conf.*, Arlington, VA., June 1982

[5] Moore, B.C., "On the Flexibility Offered in Multivariable Systems Beyond Closed Loop Eigenvalue Assignment", *IEEE Trans. Aut. Cont.* AC-21(5):689-692, Oct 1976

[6] Varga, A. "A Schur Method for Pole Assignment", *IEEE Trans. Aut. Cont.* AC-26(2):517-519, Apr 1981

[7] Konstantinov, M.M., P. Petkov, N. Christov, "A Schur Approach to the Pole Assignment Problem", *Proc. 8th IFAC World Cong.*, Kyoto, Aug 1981

[8] Klema, V.C., and A.J. Laub, "The Singular Value Decomposition: Its Computation and Some Applications", *IEEE Trans. Aut. Cont.*, AC-25(2) :164-177, Apr 1980

[9] Fletcher, R., "A New Approach to Variable Metric Algorithms", *Computer Journal*, 13:317, 1970

[10] Wonham, W.M., *Linear Multivariable Control*: *A Geometric Approach*, 2nd Edit., Springer-Verlag, New York, 1979

DESIGN AND IMPLEMENTATION OF PULSE MODULATED CONTROLLERS USING A MICROCOMPUTER

C. P. DIDUCH AND R. BALASUBRAMANIAN
University of New Brunswick
Fredericton, New Brunswick, Canada E3B 5A3

ABSTRACT

The implementation of generalized pulse modulated controllers using a microcomputer is considered. The microcomputer is configured to implement: (a) a dynamic control scheme based on output feedback, and (b) pulse modulation logic required to schedule control pulses which are functions of the sampled inputs and outputs of the process to be controlled. The microcomputer controller is general purpose in two ways: (a) it generates control pulses based on past and present controller inputs through an averaging operation of a digital filter, and (b) the pulse modulator logic is easily configured to implement standard pulse width and/or pulse frequency modulated control schemes.

INTRODUCTION

Pulse width and pulse frequency modulators find application in such diverse areas as satellite attitude control, stepper motor control, various AC and DC motor drive control, power systems conditioning and control, and communications. The importance of pulse modulated control systems lies in: (a) the inherent noise rejection properties, and (b) the simplistic nature of the actuators. In general the actuators are on in the positive or negative sense or are off. For pulse width modulated (PWM) systems the control signals are realized by periodically generated pulses of varying finite duration. Pulse frequency modulated (PFM) control signals are realized by constant duration pulses whose frequency varies.

Pulse width and pulse frequency modulators have been implemented using various forms of threshold-integrating-reset circuitry. These include the integral PFM, (IPFM),[1]; sigma PFM (ΣPFM), [2]; the lead, lag and lag-delay type PWM, [3]; and various combined pulse modulation, (CFM), schemes, [4]. The various types of pulse modulators evolved as control systems become more complex and require tighter stability margins. Facilitating control system analysis and design are a number of techniques to investigate linearized system design [5]; stability of pulse modulated control systems, [6]; and exact analysis schemes.

With the growing trend towards the used of distributed microcomputer controllers, the pulse modulator controllers which have typically used analog circuitry will be modified for use within general purpose microcomputers. The aim of this paper is to present a new scheme for implementing pulse modulator controllers using general purpose microcomputers. The discrete nature of the microcomputer lends itself naturally for the application of pulse modulation. The new schemes differ from existing schemes. Even for PWM, the modulator repetition rate need not be synchronized to the uniform input sampling rate. This is explained by the scheduling of two tasks within the microcomputer: (a) a data acquisition/digital filtering task which periodically updates a vector of buffers containing parameters of the desired control pulses, and, (b) pulse modulation tasks which accept data from the buffers and generate control pulses. Because there is no handshaking or synchronization between the tasks, both can virtually operate independent of each other in terms of frequency or repetition rate.

The only coupling between the tasks exists through an interrupt driven structure which passes the minimum amount of data to the pulse modulator logic when requested. This feature is advantageous becuase it can readily accommodate existing PWM and PFM schemes, in addition to combined pulse modulation schemes where both the pulse duration and repetition rate are varied. Such mode changing is easily accomplished in software, with a minimal change in the structure of the software, thus the controller is very flexible.

FORMULATION OF THE CONTROL PROBLEM

For convenience assume the process to be controlled is described by the normalized equations:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{1}$$

with n normalized stated, $x(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T$, r normalized outputs, $y(t) = [y_1(t), y_2(t), \ldots, y_r(t)]^T$, and m normalized inputs, $u(t) = [u_1(t), u_2(t), \ldots, u_m(t)]^T$. The equations, (1) are normalized to achieve compatability of dynamic range with the microcomputer and system interfaces.

We assume the control system has the feedback structure of Fig. 1, and that a general purpose microcomputer is implemented as the controller. The process outputs, y, and reference inputs, r, are uniformly sampled by the controller. The microcomputer is configured as a cascaded structure consisting of (a) a general purpose digital filter, and (b) a general purpose pulse modulator. In its most general form, the digital filter is described by the compatible set of equations:

$$\begin{aligned} z(kT+T) &= F\, z(kT) + G_o\, y(kT) + G_1\, r(kT) \\ v(kT) &= H\, z(kT) + L_0\, y(kT) + L_1\, r(kT) \end{aligned} \tag{2}$$

where z(kT) the digital filter state, z(kT+T) the next state, y(kT) the sampled process outputs, r(kT) the system inputs or set points, v(kT) the digital filter output vector and F, G_0, G_1, H, L_0 and L_1 are the digital filter gain matrices which are synthesized to meet desired system performance criteria.

The pulse modulator logic generates constant amplitude pulses described, at a given instant of time by three parameters, (a) the pulse duration or on-time, τ_0; (b) the off-time, τ_f; and (c) the sign of the pulse, either positive, negative or zero. Depending on the type of pulse modulator logic implemented, (either PWM, PFM, or CFM), control pulses are scheduled whose parameters are a function of the digital filter output, v(kT), where $v(kT) = (v_1(kT), v_2(kT), \ldots, v_m(kT))^T$. Specifically, assume there are m pulse modulators and the parameters of the control pulse generated by pulse modulator i are functions of $v_i(kT)$.

CONVENTIONAL PULSE MODULATION

The following is a brief description of the forms of pulse modulation that appear in the literature. Specifically, the functional relationship of the control pulse parameters and the digital filter output, $v(k)$, is outlined. In all cases, the pulse duration, $\tau_{0i}(j)$ for the j^{th} control pulse of control signal i, is a monotonic non-decreasing function of $|v_i(kT)|$, the pulse repetition rate is a non-decreasing function of $|v_i(kT)|$, and the sign of the pulse is equal to the sign of $v_i(kT)$.

A typical pulse modulated control signal, i, as shown in Fig. 2, is given by:

$$u_i(t) = \begin{cases} \operatorname{sgn}[v_i(t_{ki})], & t_{ki} \leq t < t_{ki} + \tau_{oi}(k) \\ o, & t_{ki} + \tau_{oi}(k) \leq t < t_{ki} + \tau_{oi}(k) + \tau_{fi}(k) \end{cases} \tag{3}$$

Pulse Width Modulation (PWM)

Assuming a pulse is to be generated at instant $t = t_{ki}$ the parameters of the PWM control signal are given by:

$$\tau_{oi}(k) = P_i \text{ sat } [\gamma_i |v_i(t_{ko})|] \tag{4}$$

$$\tau_{fi}(k) = P_i - \tau_o(k) \tag{5}$$

where $\operatorname{sat}(x) = \begin{cases} 1, & x \geq 1 \\ x, & |x| < 1 \\ 1, & x \leq -1 \end{cases}$ (6) and $\operatorname{sgn}(x) = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases}$ (7)

Sat is the saturation function, and γ is a constant of proportionality. The PWM control signal consists of constant amplitude pulses of positive or negative sense whose duration is proportional to the magnitude of $v_i(t_{ki})$ up to a maximum pulse width equal to the constant period, P_i, i.e., its saturated value.

Pulse Frequency Modulation (PFM)

The parameters of the PFM control signals are given by:

$$\tau_{oi}(k) = \text{constant} \tag{8}$$

$$\tau_{fi}(k) = \frac{\tau_{oi}(k)}{\operatorname{sat}[\alpha_i |v_i(t_k)|]} \tag{9}$$

where α_i is a constant of proportionality and sat is that saturation function. Define the repetition rate,

$$f_{ri} = \frac{1}{\tau_{oi}(k) + \tau_{fi}(k)} \tag{10}$$

The PFM control signal consists of constant amplitude pulses of constant duration, τ_{oi}, with sign given by sgn $[v_i(kP_i)]$ and repetition rate, f_{ri}, proportional to the saturated magnitude of $v_i(t_k)$.

Combined Pulse Modulation (CPM)

The generalized CPM control signals are given by

$$\tau_{oi}(k) = \begin{cases} \tau_{\max}, & c_1 \le |v_i(t_k)| \\ f[v_i(t_k)], & c_o < |v_i(t_k)| \le c_1 \\ o, & |v_i(t_k)| < c_0 \end{cases} \tag{11}$$

$$\tau_{fi}(k) = P_{ki} - \tau_{oi}(k) \tag{12}$$

$$P_{ki} = t_{k+1} - t_k = \begin{cases} F[v_i(t_k)], & |v_i(t_k)| > c_1 \\ T_{\min}, & |v_i(t_k)| < c_1 \end{cases} \tag{13}$$

where $f[v_k(t_k)]$ is a positive and non-decreasing over (c_0, c_1) and $F[v_k(t_k)]$ is positive and non-increasing over (c_0, c_1). In many cases CPM takes the form of PWM or PFM depending on the magnitude of $v_i(k)$. Because of the desirable large signal transients associated with PWM and the limit cycle rejection properties of PFM, the CPM controller may be chosen as a PWM controller if $|v_i(t_k)|$ exceeds some threshold say, c_t, and becomes a PFM controller is $|v_i(t_k)| < c_t$.

PULSE MODULATOR LINEARIZATION AND DESIGN PROCEDURE

If the system to be controlled is of sufficiently low pass nature, as most physical systems are, the design of the control system, i.e. selection of the digital filter gains F, G_0, G_1, H, L_0, L_1, and pulse modulator parameters, is simplified through linearization of the pulse modulator dynamics based on the equal area principle. This principle states that the response of a low pass system to a given signal, u(t), which is non-zero only in a small interval (t_0, t_1), is equivalent to that of an arbitrary signal, $u_e(t)$ provided:

$$\int_{t_0}^{t_1} u(t)\, dt = \int_{t_0}^{t_1} u_e(t)\, dt \tag{14}$$

The pulse modulator output is a non-linear function of v(kT). Making use of the equal area principle, one may replace the pulse modulator output by the signal

$$u_e(t) = G(t)\ v(t) \tag{15}$$

To simplify further, the pulse modulator logic is usually chosen such the G(t) is constant, i.e. G(t) = G. The gains G, F, G_0, G_1, H, L_0, L_1 may be synthesized using linear time invariant methods. Provided (14) holds and the output of the pulse modulator is not saturated, the pulse modulator is said to be operating in its quasilinear region. Under these conditions, the digital filter output equation and linearized pulse modulator gain may be combined as

$$u(kT) = G[Hz(kT) + L_0 y(kT) + L_1 r(kT)] \tag{20}$$

For design purposes, the continuous process dynamics may be discretized as:

$$\begin{aligned} x(kT + T) &= \phi\, x(kT) + \alpha\, u(kT) \\ y(kT) &= C\, x(kT) \end{aligned} \tag{21}$$

The linearized design can be done in a number of conventional ways. If all process states, x(t) are available for measurement, then a statefeedback controller has the structure,

$$u(kT) = GL_0\, x(kT) \tag{22}$$

where the statefeedback gain GL_0 may be chosen using LQG theory or pole-placement algorithms. If all process states are unavailable for measurement an observer based control structure may be used. A full order observer is given by:

$$\begin{aligned} z(kT + T) &= \phi\, z(kT) + \alpha\, u(kT) + K_0[y(kT) - Cz(kT)] \\ u(kT) &= GHz(kT) \end{aligned} \tag{23}$$

where, z(kT), are the estimates of the process states, K_0 is the observer gain and GH is the estimated statefeedback gain. Another possible structure is the conventional PID controller which has the form:

$$\begin{aligned} u(kT) &= k_p\, e(kT) + k_d \frac{e(kT) - e(kT-T)}{T} + k_i\, I(kT) \\ I(kT) &= I(kT-T) + \frac{T}{2}[e(kT) + e(kT-T)] \\ e(kT) &= r(kT) - y(kT) \end{aligned} \tag{24}$$

where k_p, k_d and k_i are respectively the proportional, the derivative and the integral gains. The choice of the pulse modulator gain, G, is non-unique. Thus it can be effectively tuned by appropriately scaling H, L_0, and L_1 to ensure quasilinear operation within a predetermined dynamic range.

MICROCOMPUTER IMPLEMENTATION

Discussion of specific hardware and software to implement the generalized pulse modulated controllers follows. The software, in general, consists of a number of interrupt driven tasks which (a) initialize the controller, (b) request and accept data from A/D converters and digital sensors, (c) implement the digital filter dynamics and (d) generate the control pulses. The controller has the hardware configuration given in Figure 3, the buffer interface is given in Figure 4, and pulse generation software is given in Figure 5.

The central processing unit, CPU, uniformly samples the controller input via A/D converters and/or digital sensors. Once all of the inputs are sampled, the parameters of the desired control pulses, i.e. the on-time, τ_0, the off time τ_f, and sign are computed. This involves several matrix-vector multiplications, (2), and data transfers. Specifically there are $n \times (n + r_0 + r_1)$ multiplications required to compute $z(kT + T)$ and $m \times (n + r_0 + r_1)$ multiplications to compute $v(kT)$. Several additional arithmetic and logical operations are required to obtain the parameters of each control pulse. For a 3 MHz Intel 8085 microprocessor implemented using a second order digital filter and 8 bit integer arithmetic with single input single output combined pulse modulation, typical control system bandwidth is 200 Hz.

Real Time Implementation Results

We consider 2 applications of the generalized pulse modulator controller: (a) a laboratory model servosystem which implements an observer based digital filter and a combined pulse modulation scheme and (b) a 2-input -2-output satellite attitude control system using the digital filter to estimate rate information required for a PD control scheme and pulse logic which generates PWM control signals.

(a) Servosystem Example. The dynamics of the servo system are modelled by

$$G(s) = \frac{k}{s(s\tau + 1)}$$

The digital filter was designed using observer theory and the pulse modulation logic take the form of CPM. When the process output $y(kT)$ exceeds a threshold, say c_t, then control pulses are generated by PWM and if $y(kT)$ is less than c_t, control pulses are generated via PFM. For both modes of operation the pulse generation logic is chosen such that the linearized gain is constant for both modes of operation. The controller was implemented with an 8 bit microprocessor using an 8 bit A/D converter, 8 bit integer arithmetic, with all software compacted into 2k of memory. Typical real time transients are given inFig. 6. Note: Bandwidth of the control pulses are so much greater than that for the system output that they cannot be easily recorded on the same scale.

(b) Satellite Attitude Control System Example. The satellite altitude control system considered is a three axis stabilized geosynchronous communications satellite to be controlled via pulsed sets of jet thrusters along the roll and yaw axes. Each thruster is assumed to be able to provide a constant clockwise or counter clockwise component of torque. The third, (pitch), axis is assumed to be independantly controlled. Because of the nonorthogonal location of the jet thrusters pairs and the dynamic coupling, controller design cannot be done independantly for the roll and yaw axes. Roll and yaw attitude is uniformly sampled via the pulse modulator. The digital filter is designed to estimate roll and yaw rates through finite difference methods and these are combined with proportional terms to effect PD control in such a manner that approximate dynamic decoupling occurs. The two sets of control pulses are generated via PWM. To optimize fuel consumption a deadzone is also incorporated in the controller. Duty cycle constraints on the control pulses were also imposed. The controller was implemented using an 8-bit microprocessor with a 2-channel multiplexed A-D converter, with 8 bit-integer arithmetic. The controller was interfaced to the satellite system, simulated, in real time, on an analog computer. Typical transients are given in Fig. 7.

CONCLUSION

We present an implementation scheme for generalized pulse modulation control using a microcomputer. The structure of the controller is extremely flexible. The prefiltering allows one to design the controller using conventional linear strategies and the pulse modulator logic is easily configured to implement standard PWM, PFM and CPM. This flexibility allows one to easily incorporate practical nonlinear effects such as deadzone and duty cycle constraints.

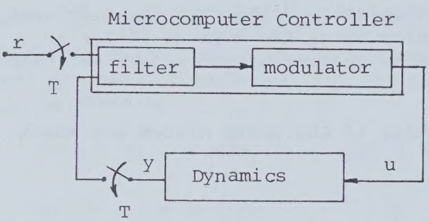

Microcomputer Controller
r
T
filter
modulator
y
Dynamics
u
T

Fig. 1 Feedback structure of the general purpose pulse modulator control system

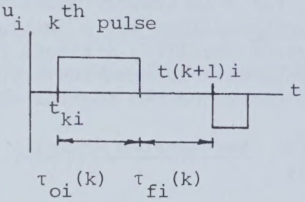

u_i
k^{th} pulse
$t_{(k+1)i}$
t
t_{ki}
$\tau_{oi}(k)$
$\tau_{fi}(k)$

Fig. 2 Typical Pulse Modulated control signal

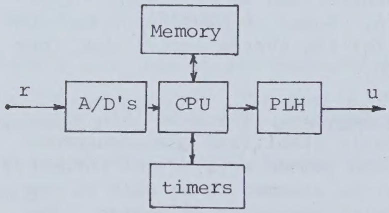

Memory
r
A/D's
CPU
PLH
u
timers

Fig. 3 Configuration of the pulse modulator controller

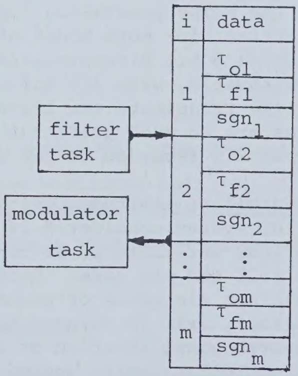

i
data
1
τ_{o1}
τ_{f1}
sgn_1
2
τ_{o2}
τ_{f2}
sgn_2
m
τ_{om}
τ_{fm}
sgn_m
filter task
modulator task

Fig. 4 The buffer interface between the concurrent pulse modulation and digital filtering tasks

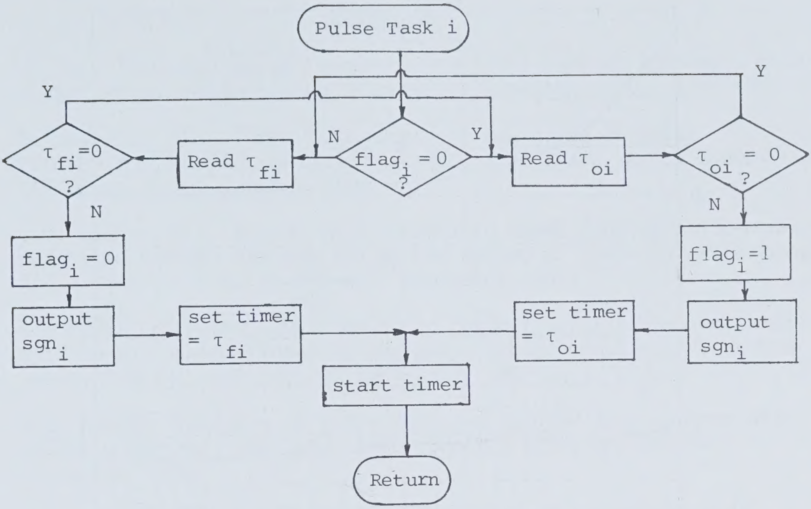

Pulse Task i
Y
Y
$\tau_{fi}=0$?
Read τ_{fi}
N
$flag_i = 0$?
Y
Read τ_{oi}
$\tau_{oi} = 0$?
N
N
$flag_i = 0$
$flag_i = 1$
output sgn_i
set timer $= \tau_{fi}$
set timer $= \tau_{oi}$
output sgn_i
start timer
Return

Fig. 5 Pulse Generation $Task_i$

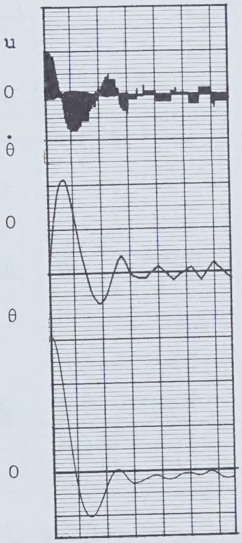


Figure 6. Transient response of the general pulse modulator servosystem, θ = output angular position.

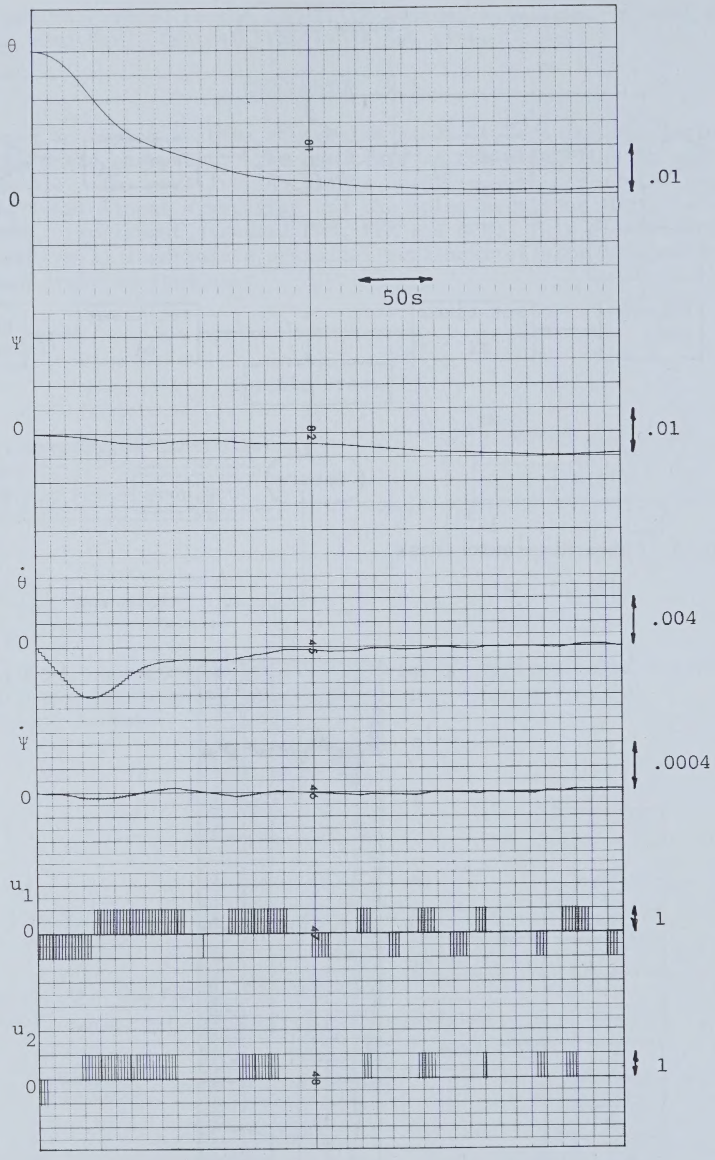


Fig. 7 Transients for a 4th order 2-input, 2 output satellite attitude control system using PWM control, T = 2s

REFERENCES

[1] C.C.Li, "Integral pulse frequency modulated control systems", Ph.D. dissertation, Northwestern University, Evanston, Ill., June 1961.

[2] T. Pavlidis, E.I. Jury, "Analysis of a new class of pulse frequency modulated feedback systems", IEEE Trans. Auto. Contr., AC-10, pp 35-43, January 1965.

[3] T.T. Kadota, H.C. Bourne,Jr., "Stability conditions of pulse-width-modulated systems through the second method of Lyapunov", IRE Trans. Auto. Contr., AC-6, pp 266-282, September 1961.

[4] V.M. Kunstevich and Y.N. Chekhovoi, "Stability of control systems with double (pulse-frequency and pulse width) modulation", Autumn Remote Control, Vol. 18, pp 1044-1052, 1967.

[5] R.E. Andeen "Analysis of pulse duration sampled data systems with linear elements", IRE Trans.Auto. Contr., AC-5, pp 306-314, September 1960.

[6] R.A. Skoog and G.L. Blankenship, "Generalized pulse-modulated feedback systems: Norma, Gains Lischitz constants and stability", IEEE Trans. Auto. Contr., AC-15, pp 300-315, June 1970.

PERFORMANCE CRITERION SENSITIVITY OF SHIP STEERING MODELS-ANALYSIS AND DESIGN IMPLICATIONS

A. K. TUĞCU, O. COŞKUNOĞLU, and R. E. REID
University of Illinois at Urbana-Champaign
Urbana, IL 61801

ABSTRACT

The problem of automatic steering control of ships in a seaway was formulated within the framework of LQG theory in a recent paper [1]. Two aspects of the formulation that are pertinent to this paper are: (i) the wave disturbances were characterized by augmenting the ship and steering dynamics with the shaping filter dynamic equation; and (ii) LQG controllers were designed to minimize a performance criterion postulated to be representative of propulsion losses due to steering.

The sensitivity of the optimal performance criterion to certain parameters of the above formulation are analyzed in this paper. The purpose of this analysis is to address the following design issues: (i) reliability of the shaping filter, (ii) robust design of observer-estimators for use as part of the compensation scheme; (iii) similarly, desensitizing the Kalman gains, and (iv) hardware tolerances. Furthermore, since the methodology given in [1] involves Monte-Carlo simulations based on nominal values of ship and environmental dynamics, the generalization of the results for perturbed values provides a partial verification of the methodology [2]. Finally, the pecuniary property of the performance index enables one to assess the consequences of small variations in the parameters in terms of a monetary unit. In these respects the sensitivity analysis in this paper is in a completely different vein than the one given by Aström [3] for normal course-keeping and the one given by Ashworth and Towill [4] for maneuvering ships.

The performance criterion gradients with respect to certain parameters are developed employing the formulas given in [5]. Sensitivity of two different controllers, each minimizing a different performance criterion, are analyzed for two ship types. The results indicate that the encounter angle of the waves and type of the ship do affect the sensitivities of the shaping filter parameters, and hence no parameter can be judged to be relatively unimportant without a full range of steering environments being considered. It was observed that the yaw disturbance shaping filter parameters are at least two orders of magnitude more sensitive than the sway disturbance shaping filter with respect to optimal cost.

REFERENCES

1. Reid, R. E., A. K. Tuğcu, and B. C. Mears, "The Use of Wave Filter Design in Kalman Filter State Estimation of the Automatic Steering Problem of a Tanker in a Seaway," Proc., 1983 American Control Conference, June 22-24, San Francisco, Ca. Also accepted for publication in IEEE Transactions on Automatic Control.

2. Ho, Y. C., "Parametric Sensitivity of a Statistical Experiment," IEEE Trans. Auto. Cont., AC-24, 982-983, 1979.

3. Aström, K. J., "Why Use Adaptive Techniques for Steering Large Tankers?," Int. J. Control, 32, 684-708, 1980.

4. Ashworth, M. J., and D. R. Towill, "Sensitivity Reduction in Ship-Maneuvering Performance via Nonliner Compensation," IEEE Proc. Part D, 129, 227-232, 1982.

5. Kurtaran, B., and M. Sidar, "Analysis of Cost Sensitivity for Linear-Quadratic Stochastic Problems with Instantaneous Output Feedback," IEEE Trans. on Aut. Cont., Vol. AC-19, 589-590, October, 1974.

MULTIVARIABLE CONTROL OF FLEXIBLE STRUCTURE - A PRELIMINARY STUDY

M. D. LODAYA
Bendix Advanced Technology Center
Columbia, MD 21045

G. G. LEININGER
Purdue University
West Lafayette, IN 47907

SUMMARY

There is a growing interest in the modeling and control of large flexible space structures. An extensive survey of current design techniques and related topics is given in Reference[1]. We investigated the control problem for the Space Construction Base Configuration I vehicle using some realistic parameters from NASA Marshall Space Flight Center. The Configuration I consisted of a central unit and two attached solar wings.

This configuration was divided into five rigid bodies connected via three degrees of freedom hinges (Figure 1) and was modeled using a discrete coordinate approach[2]. Other more accurate techniques such as finite-element analysis were not used because of their increased complexity and limited time for the study. Similarly techniques like modal cost analysis[3] and singular value decomposition were not used for model reduction for the same reasons. Instead we chose to investigate model reduction via eigenvalue analysis. Both Multivariable Nyquist Array (MNA) approach[4], and Retallack's pole placement technique[5] were used to design the controller for the Reduced Order Model (ROM). Controller performance was evaluated on a full order nonlinear model. This approach gave us some insight into the problems associated with controlling flexible structures.

The linearized three-axis model obtained using the discrete coordinate approach included all cross-coupled axis interaction and had thirty states and nine control variables. Three five-body single-axis (x, y and z) models were generated from the full-order three-axis model by neglecting cross-coupling terms. The comparison of eigenvalues computed for the full-order model and the reduced-order (single-axis) models indicated that each axis could be independently controlled. Further analysis revealed that symmetry in the torsional axis (z-axis) could simplify the control design effort. This condition does not prevail, however, in either the normal or lateral axis.

Using the symmetric single-axis model for the z-axis (six state variables and two inputs), a control system design study was performed using the MNA method and the Retallack's pole placement procedure. The MNA design method required a single loop bridged-T compensator to reduce large output oscillations. Since the bridged-T compensator is a form of closed-loop pole assignment, the remainder of the design study was performed using the Pole Placement method. Simulation results for the single-axis reduced-order model and the full-order three-axis model were identical when the pole shifts were made parallel to the real axis (active damping). For example, Figure 2 shows the angular position of body 1 (with I.C. = 0.02 rad) when the z-axis controller was implemented on the reduced-order model. A similar response is observed when the controller was implemented on the full-order model as shown in Figure 3.

A controller designed using the pole placement technique and ROM gave satisfactory performance when implemented on a nonlinear full-order model, whereas MNA did not provide a very effective controller for this structure

as it required dynamic compensators to suppress large oscillations and because of difficulty in achieving diagonal dominance for larger bandwidths.

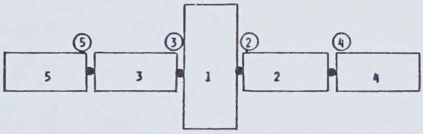
The full-scale test results confirm the negligible effects of cross-axis coupling from the torsional axis to the normal and lateral axis.

Acknowledgement

This research was supported in part by George C. Marshall Space Flight (NASA) under grant NAS8-32660.

REFERENCES

1. M. J. Balas, "Some Trends in Large Space Structure Control Theory: Fondest Hopes; Wildest Dreams," IEEE Transactions on Automatic Control, Vol. AC-27, June 1982.
2. D. P. Lipski,"Discrete Coordinate Approach to Analytical Modeling of Flexible Bodies," Internal Memorandum, Bendix Research Laboratories, Southfield, Michigan, February 22, 1979.
3. R. E. Skelton, and P. C. Hughes, "Flexible Space Structure Model Reduction and Control Design by Modal Cost Analysis," presented at 2nd AIAA/VPI symposium on the Dynamics and Control of Large Flexible Spacecraft, Blacksburg, VA., June 1979.
4. H. H. Rosenbrock, Computer-Aided Control System Design, Academic Press, London, 1974.
5. D. G. Retallack and A. G. J. MacFarlane, "Pole-Shifting Techniques for Multivariable Feedback Systems," Proceedings of the IEEE, Vol. 117, No. 5, pp. 1037-1038, May 1970.



(The individual bodies are connected to each other via three degree of freedom spring hinge suspensions. Hinge numbers are circled.)

Figure 1: Topological Diagram of SCB Configuration 1

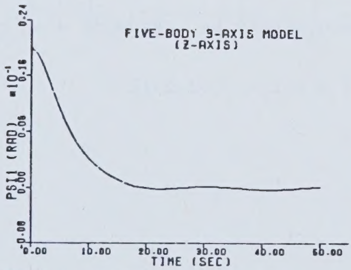


Figure 3: Angular Position Body 1

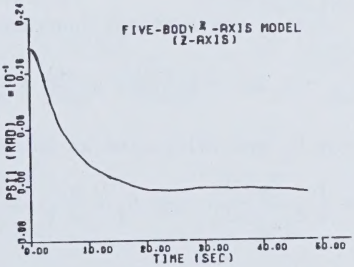


Figure 2: Angular Position (Z-Axis) Body 1

A SIMPLIFIED ALGORITHM FOR TESTING STABILITY OF DISCRETE SYSTEMS

MOHAMMED ISMAIL
Dept. of Elec. Eng., U of Nebr.
Lincoln, NE 68588-0511

HYONG K. KIM
Dept. of Elec. Eng., U of Manit.
Winnipeg, Canada R3T 2N2

ABSTRACT

An algorithm for testing stability of discrete systems, using methods in the z-domain exclusively, is described. The algorithm is based on a recently reported continued fraction method of the discrete reactance function. The new algorithm is faster than Steffen's algorithm and compares favorably with Jury's tabular form of testing stability.

INTRODUCTION

Several papers [1-4] have developed the theory of continued fraction expansion (CFE) relative to z-domain polynomials which are stable with respect to the unit circle. A polynomial decomposition $D(z) = F_1(z) + F_2(z)$ similar to the even and odd decomposition of a Hurwitz polynomial has been developed [1] where D(z) is the denominator polynomial of a stable discrete system. For a system of degree n, $F_1(z)$ and $F_2(z)$ are given by:

$$F_1(z) = 1/2\ [D(z) + z^n D(z^{-1})]\ ,\ F_2(z) = 1/2\ [D(z) - z^n D(z^{-1})] \tag{1}$$

Recently [4-5] it has been shown that the following CFE is always possible if D(z), of even degree n, has all its zeros inside the unit circle:

$$\phi(z) = \frac{F_1(z)}{F_2(z)} = R_o \frac{z-1}{z+1} + K_o \frac{z-1}{z+1}... $$

$$\phi(z) = \frac{F_1(z)}{F_2(z)} = R_o \frac{z-1}{z+1} + K_o \frac{z+1}{z-1} + \cfrac{1}{R_1 \frac{z-1}{z+1} + K_1 \frac{z+1}{z-1} + \cfrac{1}{\ddots + \cfrac{1}{R_{\frac{n-2}{2}} \frac{z-1}{z+1} + K_{\frac{n-2}{2}} \frac{z+1}{z-1}}}} \tag{2}$$

where R_i and K_i are real and positive.

The purpose of this paper is to describe an algorithm for implementing the CFE (2). The algorithm has been developed in details elsewhere [5] and has been shown to be faster than Steffen's algorithm.

The Algorithm:

Define the rational function $\phi_i(z)$ recursively by, assuming even n

$$\frac{P_i(z)}{Q_i(z)} = \phi_i(z) = R_i \frac{z-1}{z+1} + K_i \frac{z+1}{z-1} + \frac{1}{\phi_{i+1}(z)}\ , i = o,1,\ldots,(n-2)/2$$

R_i and K_i are calculated as follows [5]

$$R_i = \frac{1}{4} \frac{P_i(-1)}{P_{i+1}(-1)} \quad \text{and} \quad K_i = \frac{1}{4} \frac{P_i(+1)}{P_{i+1}(+1)}$$

where,

$$P_{i+1}(z) = \frac{Q_i(z)}{z^2-1} \quad \text{and} \quad Q_{i+1}(z) = \frac{R_{i+1}(z)}{z^2-1}$$

and

$$R_{i+1}(z) = P_i(z) - [(R_i + K_i)\, z^2 + 2\,(K_i - R_i)\, z + (R_i + K_i)]\; P_{i+1}(z)$$

Fig. 1 shows the flow diagram of computation.

Example: Consider a polynomial D(z) given by

$$D(z) = 16\, z^4 + 24\, z^3 + 18\, z^2 + 6\, z + 1$$

The computations are summarized as shown in Table 1 in which one actually needs to compute only the entries indicated in the dotted lines [5]. The new algorithm is faster than Steffen's algorithm [6] due to the fact that one is able to get two expansion coefficients instead of one coefficient in each computation cycle. For odd-ordered functions, one has to carry out the first computation cycle according to Steffen's algorithm then proceed according to the new algorithm. In terms of the required number of operations, it has been shown [5] that the new algorithm compares favorably with Jury's table form of testing stability.

References:

[1] Schüssler, H. W., IEEE Trans. ASSP-24, 1976.

[2] Szczupak et al, IEEE Trans. ASSP-25, 1977.

[3] Davis, A. M., IEEE Trans. CAS-29, 1982.

[4] Ismail, M. and Kim, H. K., IEEE Trans. CAS-30, 1983.

[5] Ismail, M. Ph.D. Dissertation, Univ. of Manitoba, Canada, 1983.

[6] Steffen, P., IEEE Trans., ASSP-25, 1977.

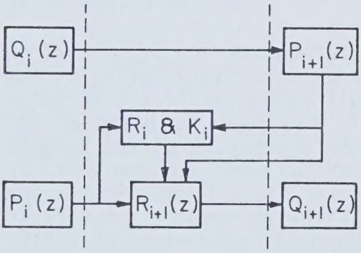
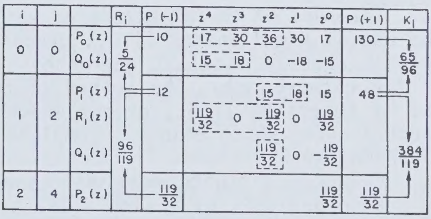


Fig. 1



i	j		R_i	P (-1)	z^4	z^3	z^2	z^1	z^0	P (+1)	K_i
0	0	$P_0(z)$		-10	17	30	36	30	17	130	
		$Q_0(z)$	$\frac{5}{24}$		15	18	0	-18	-15		$\frac{65}{96}$
1	2	$P_1(z)$		12			15	18	15	48	
		$R_1(z)$			$\frac{119}{32}$	0	$-\frac{119}{32}$	0	$\frac{119}{32}$		
		$Q_1(z)$	$\frac{96}{119}$				$\frac{119}{32}$	0	$-\frac{119}{32}$		$\frac{384}{119}$
2	4	$P_2(z)$		$\frac{119}{32}$					$\frac{119}{32}$	$\frac{119}{32}$	

Table 1

SERIAL MULTIPLE FAULT DETECTION IN BANYAN NETWORKS*

ELI OPPER and MIRO MALEK
Department of Electrical Engineering
University of Texas at Austin
Austin, Texas 78712

ABSTRACT

A serial method for multiple fault detection in a class of multistage interconnection networks is presented. The method is based on graph theoretic approach and allows simple on-line fault detection of regular rectangular banyan networks with arbitrary number of levels L and arbitrary fanout f. The method can detect, using a single port, any multiple stuck-at-0, stuck-at-1 and bridge faults on the data lines at the vertices and the edges of the network. A generalization of the method to any rectangular multistage network has also been presented.

Index Terms: Banyan network, Euler digraph, Eulerian, fault detection, fault model, multistage interconnection networks, multicomputer system.

I. INTRODUCTION

An extensive research is being currently conducted on various aspects of reconfigurable multimicrocomputer systems, based on multistage interconnection networks (e.g., see surveys [1], [2], [3]). The proper functioning of the interconnection network is very important to the overall system performance and hence an efficient fault diagnosis method for the network is necessary.

Several papers on fault diagnosis of multistage interconnection networks were recently published [4]-[11]. In this paper we present a new multiple-fault detection method based on graph theoretic approach and applicable to the general class of banyan interconnection networks [12] with arbitrary fanout f and arbitrary number of levels L. The proposed serial method tests the network from a single port (computer or processor) and it can be carried out during the normal operation in an on-line mode.

The idea of serial detection was first introduced in [10]. Based on the use of Hamiltonians (i.e., graphs which traverse every vertex of the network exactly once) it guarantees the fault detection in the vertices of the network only. The method proposed in this paper is based on the use of Eulerians (i.e., graphs which traverse every edge of the network exactly once) which enable a complete cover of the edges and the vertices of the network.

Eulerian circuits were discussed in [13] with respect to a fault tolerance analysis of a simple 2x2 crossbar switch based networks called β networks. In [13] the analysis assumed that the faulty elements has been already detected and located and used the Eulerian circuits to verify a fault tolerance property in the faulty β network. Here the Eulerian circuits serve as a basis for a serial detection method to be used in a general class of networks.

In Section II we give a short description of banyan interconnection networks, basic definitions and the fault model. In Section III we develop the serial fault detection method and in Section IV we generalize it. Concluding remarks are given in Section V.

---

*This research was sponsored in part by the IBM Corporation.

II. PRELIMINARIES

A. Banyan Topology

The results presented in this paper are mainly based on SW-banyan properties. Hence we first give a brief review of banyan terminology.More detailed description was given in several papers (e.g., [12], [14]).

Banyan: A Hasse diagram [15] of a partial ordering in which there is one and only one path from any base to any apex.

Base: Any vertex with no arcs incident into it.

Apex: Any vertex with no arcs incident out of it.

L-level banyan: A banyan in which every path from any base to any apex is of length L.

Fanout (f): The out-degree of a vertex.

Spread (s): The in-degree of a vertex.

(s,f,L) regular banyan: An L-level banyan in which the in-degree of every vertex except the bases is s and the out-degree of every vertex, except the apexes is f.

(f,L) regular rectangular banyan: An (s,f,L) regular banyan in which f=s.

(s,f,L) regular SW-banyan: An (s,f,L) regular banyan in which for any pair of bases, if their correspondence sets of any degree are not disjoint, then they are identical and for any pair of apexes, if their inverse correspondence sets of any degree are not disjoint, then they are identical. The above axiomatic definition is given in [14]. The more common definition is a constructive one which defines SW-banyan as a crossbar or a collection of crossbars interconnected in a pattern characterized by an SW-banyan interconnection graph.

(f,L) regular rectangular SW-banyan: An (s,f,L) regular SW-banyan in which f=s. An example of a (3,2) regular rectangular SW-banyan is shown in Figure 1.

Notice that the banyan graph is defined as a directed graph from bases to apexes. But according to [12] it can also be defined as a directed graph from apexes to bases while preserving all banyan properties.

B. The System Models

Two system models are considered in this paper. In the first (called system (f,L) type 1) each one of f^L computers (processors with local memories) is attached to both sides of a regular rectangular SW-banyan network with fanout f and L levels. The communication through the network is unidirectional and the control is based on the routing tag technique. An example of system (3,2) type 1 is shown in Figure 2. In the second model (called system (f,L) type 2) f^L processors are attached to one side of the network (apexes) and f^L memories are attached to the other (bases). The communication through the network is bidirectional via shared memories and the control is based on the routing tag technique. An example of system (3,2) type 2 is shown in Figure 3. The detailed labeling and routing scheme used in the above models is not given here but can be found in several references (e.g., [10]).

We want to emphasize that in the discussed models each edge and vertex represent a collection of lines (wires) which basically include the data, memory address, control lines and routing tags (source and destination). Throughout this paper we assume that the data part at each vertex and edge contains k lines (typical values vary from k=8 to k=60).

C. Basic Definitions

We present here several basic definitions which are useful throughout this paper.

Test graph: A subgraph of the network graph which is constructed (mapped) to accomplish a test(s). In other words, it is a subgraph of an interconnection graph which "reserves" paths for test signal traversal. In specific cases we use the notations test path, Hamiltonian test graph and Eulerian test graph.

Testing packet: A packet generated by a computer (processor) and transmitted via a path(s) in a test graph for the purpose of testing. A testing packet can be distinguished from other message packets by a special test flag.

In the diagnosis method developed in this paper, two distinct testing packets are used. A packet with alternate 0's and 1's, starting with 0, at its data lines and the complemented packet with alternating 1's and 0's, starting with 1. Throughout this paper we use the brief notation "01" and "10" for the above testing packets, respectively.

Test: Generation and traversal of one testing packet from one several or all the computers (processors) through a test graph and validity verification. In other words, a test is composed of the generation of a vector (pattern) of testing packets, its traversal along a test graph and validity verification.

Serial diagnosis: Diagnosis method in which a single computer (processor) generates and verifies testing packets.

D. The Fault Model

In this paper, permanent multiple faults of the following types are considered:

Stuck-at faults: s-a-0 and s-a-1 (written for brevity as s-a-x) faults on the data lines at the edges and the vertices of the network graph.

Bridge faults: AND bridges and OR bridges between pairs of adjacent data lines embedded at the same edge or vertex of the network graph (where a bridge is a shorted pair of lines with the combined effect of logical AND or OR).

The above fault model is presented with respect to the edges and the vertices of the network graph. As a result the fault detection method described in this paper is basically implementation independent and can be applied to all the networks which can be represented by a similar graph.

An example of interpretation to the graph is the one commonly used with respect to the banyan networks. In this case a vertex represents a bus and an edge represents a switch contact. If, for example, in an actual real system each IC is composed of one fxf crossbar switch module, a vertex is further interpreted as an external bus (i.e., the lines outside the IC as well as its pins) and an edge is interpreted as an internal bus (i.e., the lines inside the IC).

III. DISCUSSION

The proposed serial method is based on the construction of a particular test graphs (e.g., Eulerians) which traverse all the edges and the vertices of the network in an efficient way. The method is general and can be applied to networks with arbitrary fanout f and arbitrary number of levels L.

Lemma 1: In a system (f,L) type 1 a test graph can always be constructed in such a way that it traverses, exactly once, all the edges of the network graph starting and ending at the same vertex.

Proof: Let us delete all the computers from a system (f,L) type 1 representation and merge each apex i with the corresponding base i for all $0 \leq i \leq f^L-1$. The resulting graph, an (f,L) regular rectangular banyan with merged apexes and bases, is a connected balanced digraph and hence it is an Euler digraph [16]. An algorithm to construct an Euler line in any Euler digraph from a given rooted spanning tree is also described in detail in [16]. The construction of a rooted spanning tree in a banyan is a trivial problem due to its basic structure (i.e., banyan can be viewed as a set of overlapping trees). If we use an Euler line as a test graph in a system (f,L) type 1 any edge of the network is traversed exactly once by definition. □

Figure 4 shows an example of an Eulerian test graph in a system (3,2) type 1. In this example the numbers at the edges show the traversing order along the Eulerian starting and ending at c(0).

By using an Eulerian test graph in a system (f,L) type 1, each computer and the links that connect it to the network (i.e., to the corresponding apex and base), are traversed exactly f times as well as the vertices of the network. However, from graph theory point of view the computers and their connecting links to the network can be thought of as embedded in the corresponding merged vertices. Thus preserving the definition of Eulerian for the complete system.

Notice that the β graph defined in [13] is *not* isomorphic to the special case of a (2,L) regular rectangular SW-banyan. As a result the proof of Lemma 1 is not exactly the same as the corresponding proof in [13] even though both are based on the same well-known result about Euler digraphs.

Theorem 1: Only *two* testing packets, generated and verified by the same single computer, are needed to detect a single s-a-x or bridge fault at any edge or vertex of a system (f,L) type 1 *independent* of the fanout f and the number of levels L.

Proof: By Lemma 1, an Eulerian test graph can be constructed starting and ending at any computer c(i) for $0 \leq i \leq f^L-1$. If that computer transmits the two testing packets "01" followed by "10" or vice-versa, a s-a-x fault along the Eulerian test graph is detected at c(i) by observing the same value x on the corresponding stuck line for the both testing packets. A bridge fault along the Eulerian test graph is detected at c(i) by observing the same value, 00 for AND bridge and 11 for OR bridge, on two adjacent lines for each one of the two testing packets. □

Lemma 2: In a system (f,L) type 2 a test graph can always be constructed in such a way that it traverses, exactly once (in each direction), all the edges of the network graph starting and ending at the same vertex.

Proof: The bidirectional network of a system (f,L) type 2 can be described as two back-to-back unidirectional (f,L) regular rectangular banyans with the apexes connected through the memories and the bases through the processors. If we merge the corresponding bases, after deleting the memories and the corresponding apexes, after deleting the processors,the resulting merged graph is clearly a connected balanced digraph. The end of the proof follows the proof of Lemma 1. □

Figure 5 shows an example of an Eulerian test graph in a system (2,2) type 2.

Theorem 2: Only *two* testing packets, generated and verified by the same *single processsor* are needed to detect a single s-a-x or a bridge fault at any edge or vertex of a system (f,L) type 2 *independent* of the fanout f and the number of levels L.

Proof: Similar to the proof of Theorem 1. □

Corollary 1: The proposed serial detection method can detect any combination of multiple s-a-x and bridge faults.

Proof: As the two testing packets traverse along the Eulerian test graph, faults can mask each other (e.g., s-a-1 can mask s-a-0 on the same bit) but it cannot compensate each other. Hence the last fault in the traversing order along the Eulerian test graph is always observed with respect to a particular bit or a pair of adjacent bits and any combination of multiple faults is thus detected. □

The serial detection method can be used in two basic modes of operation:

Off-line mode: In this mode the detection method is carried out while the normal operation is stopped. It can be executed before and periodically during the normal operation. The network can be buffered or unbuffered. A modification for circuit switching is also possible in an off-line mode.

On-line mode: In this mode the detection method is carried out continuously throughout the normal operation. The testing packets traverse the network as any other message packet and the fault detection is done as soon as a testing packet returns to the testing computer (processor). It should be noticed that more than two testing packets can traverse the network simultaneously using alternate packets of "01" and "10". The fault detection is done by verification of each pair of consecutive testing packets. This feature is useful especially in a very large system (i.e., f^L is large) and where real time detection is of importance. The number of testing packets traversing the network simultaneously can be varied in time and should be decided due to the actual traffic, the network reliability and the desired testing performance. A good performance of the method in an on-line mode is achieved with buffered networks. In unbuffered networks a testing packet can be deleted as a result of a conflict and the efficiency of the method can be reduced.

The serial detection method is optimal in the sense that all the elements under test (the edges and the vertices) are covered, while each element is traversed the minimum number of times.

Since the Eulerian test graphs traverse all the f^L computers (processors) in a system (f,L) the serial detection method is not bound to any particular computer (processor) and each one of the f^L computers (processors) can generate and verify the testing packets, thus enhancing the flexibility of the method.

It should be understood that in a real system faults on the 'non-data' lines (i.e., control, tags, memory address) may lead to one of the following:

1) Creation of a non-Eulerian circuit starting and ending at the original testing computer (processor) as a result of misrouting.
2) Creation of an inner ('endless') circuit as a result of misrouting.
3) Loss of the testing packets (breaking the Eulerian) as a result of control faults.
4) Non-testing packet traversal along the Eulerian as a result of memory address faults.

The occurence of 1) can be detected by checking the source (if available) of the testing packet by each computer (processor) or by dynamic counting of the circuit length. Time-out mechanism may handle 2) and 3) and the occurence of 4) is detected by checking the test flag of each testing packet and by detection of faulty data. We want to point out that the above measures though adding some 'non-data' lines fault detection capability to the serial method do not guarantee a complete testing of those lines. A complete detection cover of faults at the memory address lines along the network can be achieved by using the same pattern of alternating 0's and 1's as for the data lines. In this case the corresponding two memory addresses (i.e., 0101,... and 1010...) in each memory module must be preserved for the testing procedure.

In some systems a packet is composed of several (e.g., 8) bytes or words which are serially time multiplexed on the same lines while the routing tag and the memory address are embedded in the leading byte(s). In such systems since there is no separation among data lines, memory address lines and routing tag lines, the proposed serial detection method covers all the corresponding faults.

IV. GENERALIZATION

Corollary 2: The serial detection method can be applied to any multicomputer system configured as a system (f,L) type 1 or a system (f,L) type 2 and based on a regular rectangular multistage interconnection network.

Proof: A regular rectangular interconnection network can be described as a connected balanced digraph in the same manner as was shown in the proof of Lemma 1 and Lemma 2. Hence, by Theorem 1 and Theorem 2 an Eulerian test graph can always be constructed and the serial detection method can be applied. □

Corollary 2 implies that the proposed serial detection method can be applied to the majority of the multistage interconnection networks that are now under research and development. This includes the baseline and all its topologically equivalent networks [17], the rectangular CC-banyan [12], the Beneš network [18], and others.

The proposed method can be further generalized for use in any computer network described by a connected balanced digraph. In such a network an Eulerian test graph can always be constructed and the detection method can be applied efficiently.

V. CONCLUDING REMARKS

A graph theory based method for fault detection in a large class of multistage interconnection networks has been presented. The method allows multiple fault detection in rectangular banyans and equivalent networks with arbitrary number of levels L and arbitrary fanout f. The method is simple to implement and it is optimal in the sense that each edge is traversed exactly once by the testing data which is generated and verified by a single computer (processor) only. The method is adequate, especially for on-line fault detection in packet switching networks, and can serve as a first phase of a complete fault diagnosis of the network. The described serial fault detection method, though shown for multistage interconnection networks, is more general and can be applied to any computer network that can be represented by a connected balanced graph. The method is especially applicable to VLSI due to a very low pinout requirement.

REFERENCES

[1] H.J. Siegel, R.J. McMillen and P.T. Mueller, "A survey of interconnection methods for reconfigurable parallel processing systems," AFIPS Conf. Proc., vol. 48, June 1979, pp. 529-542.

[2] T.Y. Feng, "A survey of interconnection networks," Computer, Vol. 14, No. 12, Dec. 1981, pp. 12-27.

[3] G.J. Lipovski and M. Malek, "A theory for multicomputer interconnection networks," Technical Report TRAC-40, The University of Texas at Austin, March 1981.

[4] K.M. So and J.J. Narraway, "On-line fault diagnosis of switching networks," IEEE Trans. Circuit Systems, Vol. CAS-26, No. 7, July 1979, pp. 575-583.

[5] B.D. Rathi and M. Malek, "Fault diagnosis of interconnection networks," Proc. of Data Acquisition, Computing and Control Symp., Dec. 1980, pp. 110-119.

[6] K.M. Falavarjani and D.K. Pradhan, "Fault diagnosis of parallel processor interconnection networks," Digest of Papers 11th Fault-Tolerant Computing Symp., June 1981, pp. 209-212.

[7] T.Y. Feng and C.L. Wu, "Fault diagnosis for a class of multistage interconnection networks," IEEE Trans. Computers, Vol. C-30, No. 10, Oct. 1981, pp. 743-758.

[8] D.P. Agrawal, "Testing and fault tolerance of multistage interconnection networks," Computer, Vol. 15, No. 4, April 1982, pp. 41-53.

[9] W.Y.P. Lim, "A test strategy for packet switching networks," Proc. of the 1982 International Conference on Parallel Processing, Aug. 1982, pp. 96-98.

[10] E. Opper and M. Malek, "Real time diagnosis of banyan networks," Proc. of the Real-Time Systems Symp., Dec. 1982, pp. 27-36.

[11] M. Malek and E. Opper, "Multiple fault diagnosis of SW-banyan networks," Digest of Papers 13th Fault-Tolerant Computing Symp., June 1983, pp. 446-449.

[12] L.K. Goke and G.J. Lipovski, "Banyan networks for partitioning multiprocessor systems," Proc. of First Ann. Symp. on Computer Architecture, Dec. 1973, pp. 21-28.

[13] J.P. Shen and J.P. Hayes, "Fault tolerance of a class of connecting networks," Proc. 7th Ann. Symp. on Computer Architecture, pp. 61-71, 1980.

[14] U.V. Premkumar, "A theoretical basis for the analysis and partitioning of regular SW-banyans," Ph.D. Diss., The University of Texas at Austin, 1981.

[15] C. Berge, The Theory of Graphs and its Applications, John Wiley and Sons, Inc., New York, 1962.

[16] N. Deo, "Graph Theory with Applications to Engineering and Computer Science," Prentice-Hall, Inc. N.J. 1974.

[17] C.L. Wu and T.Y. Feng, "On a class of multistage interconnection networks," IEEE Trans. Computers, Vol. C-29, August 1980, pp. 694-702.

[18] V.E. Beneš, "Mathematical Theory of Connecting Networks and Telephone Traffic" Academic Press, New York, 1965.

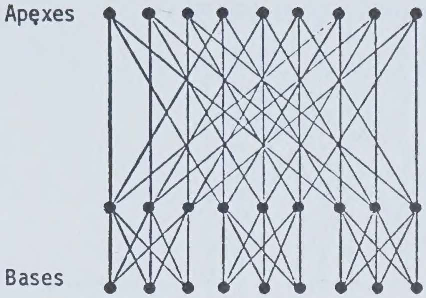

Apexes
Bases

Figure 1. Rectangular banyan with f=3 and L=2.

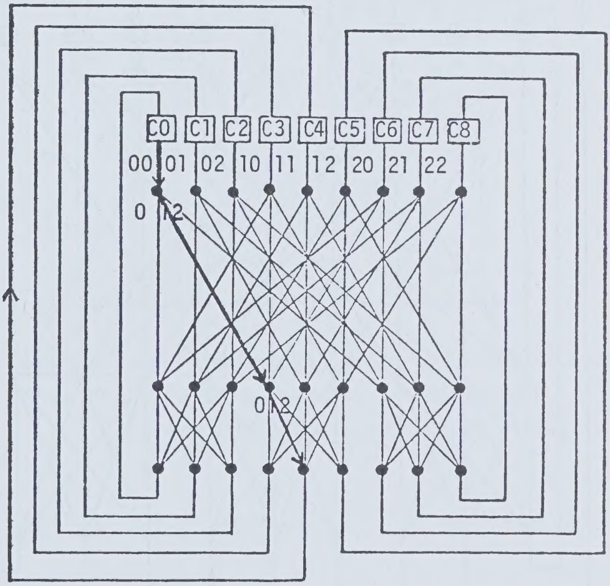

C0
C1
C2
C3
C4
C5
C6
C7
C8
00
01
02
10
11
12
20
21
22
0 1 2
0 1 2

Figure 2. System (3,2) type 1 with an example of the routing from C0 to C4.

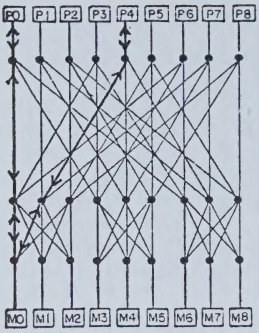

P0
P1
P2
P3
P4
P5
P6
P7
P8
M0
M1
M2
M3
M4
M5
M6
M7
M8

Figure 3. System (3,2) type 2. The example shows how processors P0 and P4 communicate via shared memory M0.

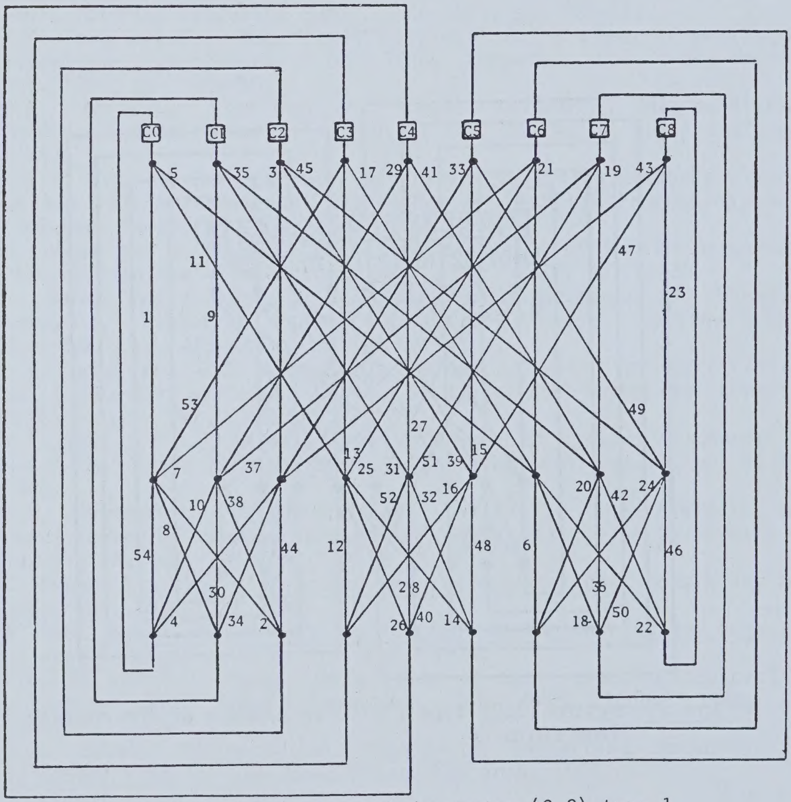

C0
C1
C2
C3
C4
C5
C6
C7
C8
5
35
3
45
17
29
41
33
21
19
43
11
47
23
1
9
53
49
27
13
25
31
51
39
15
16
7
37
10
38
52
32
20
42
24
8
54
44
12
48
6
46
30
28
36
50
4
34
2
26
40
14
18
22

Figure 4. An Eulerian test graph in system (3,2) type 1. (The numbers indicate the traversing order starting and ending at C0).

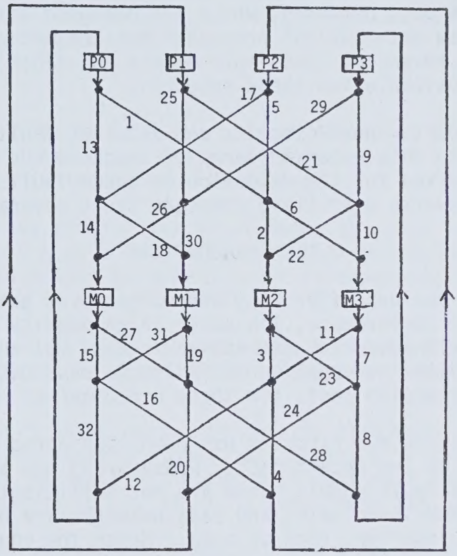

P0
P1
P2
P3
25
17
5
29
1
13
21
9
26
6
14
30
2
10
18
22
M0
M1
M2
M3
27
31
11
15
19
3
7
23
16
24
32
8
28
20
12
4

Figure 5. An Eulerian test graph in System (2,2) type 2. (The numbers indicate the traversing order starting and ending at P0).

TEST DERIVATION FOR MOS SWITCH LOGIC NETWORKS

DAMODARAN RADHAKRISHNAN
Department of Electrical Engineering
Old Dominion University
Norfolk, VA 23508

GARY K. MAKI
Department of Electrical Engineering
University of Idaho
Moscow, ID 83843

ABSTRACT

Testable design of digital chips is a major concern of LSI and VLSI designers. High performance, low power consumption and high circuit densities are available with pass networks, which are designed with minimum sized transistors. A new mathematical procedure for the design of pass networks has been recently advanced. But fault models and fault detection techniques are not readily available for these networks.

The open-circuit, short-circuit, and stuck-at faults in the pass network are modeled in this paper in terms of stuck-on and stuck-open faults. Conditions are derived for the detection of these faults. In addition, the design of pass networks exhibiting complete fault coverage is presented.

I. INTRODUCTION

Switch logic designs offer many advantages over gate logic networks. These include high performance, low power, high density and minimum size [1]. These features make them attractive in many LSI and VLSI designs. This paper deals with the modeling of faults in pass networks and the derivation of fault detection tests for these circuits.

Fault modeling in MOS circuits has been considered by many authors [2, 3, 4, 5, 6, 7, 8] in the recent past. Wadsack [5] has shown that the conventional stuck-at fault models alone are not sufficient for the modeling of MOS faults. CMOS logic gates and pass networks are basically tristate logic devices - output low, high or open. Hence, the conventional stuck-at fault models must be modified to include the tristate property of these devices. Two notions, stuck-open (s-op) and stuck-on (s-on) faults were introduced by Wadsack [5]. The s-op and s-on fault models for MOS circuits describe many of the fault conditions more accurately. A fault due to a Source-Drain short or a s-a-1 value on the control Gate of a MOS Transistor can be described by a s-on fault. Similarly, a s-op fault describes the s-a-o or a missing connection to the Gate of individual FET's, an open conductor, or missing connection to either the Source or Drain of an FET. However, for this fault to be considered as a true "open" any residual capacitive or resistive coupling must be negligibly small.

The s-op faults may introduce memory within an otherwise combinational circuit by retaining its previous value for a time period determined by the time constant of the network. Hence, a purely combinational network under s-op fault conditions may exhibit sequential behavior. Fault detection tests must therefore be derived taking the above sequential nature of these networks into consideration.

II. PASS NETWORK DESIGN

A pass network is an interconnection of a set of pass transistors to achieve a particular switching function. The design of these networks has been an art that in the past was left to the cleverness of the logic designer because of the absence of any mathematical formulation to define their input/output relationship. A new mathematical procedure for designing pass networks has been recently advanced [9, 10]. A brief review of this procedure is given here to establish the basis for the derivation of fault detection tests. A simple NMOS pass transistor gate and its truth table are shown in Figure 1. When the control input is held high, the input is passed to the output, else the output is held floating. The variables which control the gates of the individual transistors are called control variables and the variables which feed the inputs are called pass variables. A generalized pass network is shown in Figure 2. Each Pi is composed of a series of transistors that when enabled simultaneously present Vi to the output f. Hence, Pi can be represented as a product term, where each literal of Pi represents a pass transistor. The output pass function f can be represented as:

$$f = P1(V1) + P2(V2) + \dots + Pn(Vn).$$

When Pi = 1, f = Vi.

Definition 1: If Vi denotes the pass variable and Pi denotes the product term, then Pi(Vi) is called the pass implicant; Vi is in the set {0, 1, Xi, Xi'} where Xi is an input variable.

In contrast to the traditional implicants, the pass implicants represent either the 0's, 1's or a combination of both 0's and 1's. The design of a pass network can easily be done using a Karnangh map [9, 10]. The minimization procedure is summarized below.

1) Each and every entry in the map must be covered at least once.
2) Pass implicants and pass variables are identified in accordance with the following:
    a) Each implicant consists of 2^i cells that have the identical relationship as traditional implicants.
    b) The pass variable Vi in the implicant must be equal to one of the elements {0, 1, Xi, Xi'} throughout the implicant.

An example to illustrate the above procedure is shown in Figure 3. Three pass prime implicants will cover the whole map. Hence, the pass function becomes

$$f = A'B'(D') + AB'(C') + B(D)$$

This function can be realized as shown in Figure 4. A reduction in the transistor count is possible by factoring B' in the above function, thus yielding

$$f = B'(A'(D') + A(C')) + B(D)$$

III. FAULT MODELS FOR PASS NETWORKS

The faults in a pass network can be classified as s-a-faults and open and short circuit faults in the pass transistors. The short circuit and open circuit faults can be modeled as s-on and s-op faults respectively.

A. Stuck-at-faults

The different s-a-faults occurring are s-a-0, s-a-1, s-op, and s-on faults in the transistor.

A s-a-1 fault in the control variable will always keep the transistor 'ON', thus shorting the input and output leads of the transistor. Thus, when Pi is s-a-1 in Figure 1, Z = 1(Vi). It may happen that a conflict as a result of the s-on fault occurs at the output due to this fault [6]. Thus, in general, a s-a-1 fault in the control variable may be modeled as a s-on fault.

A s-a-0 fault in the control variable turns the transistor 'OFF', and hence the output of the gate will be in the high impedance state, Z=0(Vi). The transistor behaves like an open circuit making this fault equivalent to a s-op fault.

A s-a-1 fault in the pass variable provides a constant 1 at the output of the gate independent of the state of the pass variable. Similarly, a s-a-0 fault provides a constant 0 at the output. A s-op fault in the pass variable will have the same effect as a s-op fault in the control variable.

B. Short circuit faults

Three different kinds of shorts may occur in a pass transistor. They are Gate-Source short, Source-Drain short, and Gate-Drain short.

A Gate-Source short produces a high impedance state at the output whenever a 0 is to be passed to the output. Under all other conditions the gate works satisfactorily. So this fault may be included under the s-op fault. From Figure 1 it is easily seen that this fault may affect other nodes of the circuit by feeding a 0 for the pass variable Vi whenever the control variable Pi is 0.

Since a Drain-Source short always passes the pass variable to the output, it is equivalent to a s-on fault in the control lead.

Under the Gate-Drain short, when the control input becomes 0, a conflict may occur if a 1 is passed to the output by some other implicant. In addition, this may behave like a s-op fault in the control lead when a 0 is to be passed.

C. Open circuit faults

Open circuit faults produce a high impedance output whether the fault occurs in the control or pass variables of the circuit and hence behave like s-op faults.

Thus, in general, all faults in a pass transistor can be classified into three categories: s-a-faults, s-op faults, and s-on faults. The control variable failures are s-on and s-op, whereas the pass variables can assume all three cases.

IV. FAULT DETECTION IN PASS NETWORKS

A. Pass variable faults

Consider the pass network:

$$Z = P1(V1) + P2(V2) +..+ Pi(Vi) +..+ Pj(Vj) +..+ Pn(Vn) \qquad (1)$$

Any pair of product terms Pi and Pj will always differ at least in one variable. This makes it possible to sensitize a path for a pass variable to the output in many networks.

Lemma 1: The set of tests for the detection of pass variable faults at the primary inputs in a pass network also detects all the faults in the internal nodes of the path connecting the pass variable to the output.

Proof: Let Pi=1 in Eq. (1) which makes Z=Vi. From the nature of a pass network, there is a single conducting path from the output Z to the input pass variable Vi. It is then clear that a fault along the path connecting Vi to Z is equivalent to a fault at the input Vi.

Q.E.D.

Theorem 1: For a pass network, a s-a-1 or a s-a-0 fault in a pass variable Vi is detectable if a path from this variable to the output of the network can be sensitized and no two product terms Pi and Pj are true simultaneously.

Proof: To detect a fault in a pass variable Vi, it is necessary to sensitize a path for Vi to the output. This is possible only by making Pi=1. When this fault is being sensitized, if Pj=1, then both the variables Vi and Vj are passed to the output and hence this fault cannot be detected. If Pi=1 and all Pj=0, then fault in Vi is detectable since output Z=Vi.

Q.E.D.

Theorem 1 gives necessary conditions for the generation of tests for s-a-faults in the pass variables. The set of tests is given by the following:

Fault	Test
Vi s-a-0	Vi Pi
Vi s-a-1	Vi' Pi

This is illustrated in the following example.

Example 1: Consider the pass function:

Z = B(D) + B'(A(C') + A'(D'))

The pass network for this function is shown in Figure 5.

Let Pi(Vi) = B(D). Using Theorem 1, to test for s-a-faults in D, enable Pi, i.e., Pi=1 and disable all other Pj's. Hence the test for D s-a-1(0) is B=1 and D=0(1). Similarly, the test for s-a-1(0) fault in D' is A=0, B=0 and D=1(0). The tests for other pass variables may be found in a similar manner.

The following example illustrates a case where the conditions of Theorem 1 are not met.

Example 2. Consider the pass function:

Z = AB'(C') + B(A') + A'(1)

The pass network is shown in Figure 6. A s-a-0 fault in the pass variable A' cannot be detected since the sensitization of this variable to the output also activates the implicant A'(1).

B. Control variable faults

Definition 2: In a pass network, let two product terms Pi and Pj be equal to 1 simultaneously, where Vi<>Vj. If the output assumes the value Vi(Vj), then Vi(Vj) is said to dominate Vj(Vi).

The dominance property is equivalent to the logical strength relationship defined by Hayes [4]. By the above definition of dominance, a '0' always dominates a '1' as long as the pass variable '1' is fed from VDD through a depletion mode transistor.

Theorem 2: For a product term Pi = X1X2...Xi...Xn of a pass network, a s-on fault in Xi is detectable if:

(a) at least one product term Pj<> Pi can be found such that Pj=1 whenever X1X2...Xi...Xn are assigned 11...0...1(Xi=0), and

(b) Vi dominates Vj for all those Pj for which Pj=1 under the above logic assignment.

Proof: To sensitize a path to test for control variable Xi s-on, Xi must be 0 and all the other variables in the product term Pi are assigned 1. Thus under fault conditions the pass variable Vi is passed to the output. If at least one Pj=1 for this logic assignment, then Vj is also passed to the output. If logic assignments can be made such that Vi dominates all Vj for which Pj=1, then the output will be Vi under s-on fault conditions on Xi. On the other hand, under fault-free conditions, Pi=0 and hence output equals Vj. Thus, s-on fault on Xi is detectable.

Q.E.D.

Since each of the above fault conditions produces a false output value, they can be detected with the application of a single test vector. The next class of faults produces a high impedance output which the following discussion will show is much more difficult to detect. Consider a case where the input sensitizes a fault such that the output Z is in the high impedance state. The only way this condition can be detected without changing the input is through the use of a "high impedance detector". This device would involve a pull-up or pull-down depletion mode transistor which would force the output to a specific state. The presence of this device would consume significant power and increase the propagation delay. For these reasons, this approach is not attractive.

A more pleasing approach to detecting a high impedance output is to use a pair of input test vectors, which produce complementary output states. Let these vectors enable Pi(Vi) and Pj(Vj), and assume a s-op fault has affected Pj. Application of the two inputs would produce outputs Vi, Vi'. If a s-op fault affects Pj, the output is Vi, high impedance; the output circuit would interpret the output as Vi because of charge stored in the output capacitance. Clearly, the fault is detectable.

The following theorem deals with finding the test vectors for s-op faults using the two vector approach.

Theorem 3: A s-op fault in the control variable Xi, occurring in the product term Pi of a pass network, can be detected by a sequence of test vectors Ij, Ii which enable Pj(Vj) and Pi(Vi) respectively if:

1) for the input Ij, Pj=1 and Pi=0.
2) for input Ii, Xi=1, Pj=0 and Pi=1 under fault-free conditions.
3) the value of Vj can be assigned to be the complement of Vi.

Proof: For Xi s-op, under all logic assignments of the variables in the pass network, Pi=0 and hence the pass variable Vi is never passed to the output. The problem is to discover a means to detect a high impedance output. One way is to force the output to undergo a change of state when Pi is assigned to 1. This occurs as follows:

1) Apply an input vector such that Vj is passed through Pj to the output of the network.
2) Apply input vector such that Pi=1 with Vj' as the output. If Xi is s-op, then the output will maintain value Vj. If Xi is fault-free, then the output will assume Vi and the output will undergo a transition proving that Pi is not s-op.

Q.E.D.

Example 3 illustrates the detection of control variable faults in a pass network.

Example 3. Consider the pass function in Example 1.

Z = B(D) + B'(A(C') + A'(D'))

Let Pi(Vi) = B(D). Using Theorem 2 and the assumption that a '0' dominates a '1', the test for 'B' s-on is:

(ABCD) = (1000) or (00-0)

Now let Pj(Vj) = AB'(C'). Using Theorem 3, the sequence of test vectors to detect 'B' s-op is: (ABCD) = (100-, -1-0).

V. TESTABLE PASS NETWORKS

The theorems presented earlier provide means by which fault detection tests can be derived for a general pass network. As illustrated in Example 2, it is not possible to find a complete test set for all networks. The networks to be described below exhibit good fault detection properties compared to other networks.

A. Disjoint Pass Networks

Definition 3: A pass network with disjoint pass implicants is called a disjoint pass network. In a disjoint pass network, when any one product term Pi=1, all the others are 0.

Corollary 1: All faults in pass variables are detectable.

Proof: Since no two product terms Pi and Pj are ever true simultaneously for a given input condition, the conditions of Theorem 1 are met. The test for Vi s-a-1(0) is:

set Pi=1 and Vi=0(1) for every i.

Q.E.D.

Corollary 2: A s-on fault in the control variable Xi in the product term Pi of a network is detectable if Vi dominates Vj, where Pj=1, whenever the variables in Pi are assigned logic values so as to sensitize a path for Vi to the output with Xi=0.

Proof: This follows directly from Theorem 2.

Corollary 3: A s-op fault in the control variable Xi, occurring in the product term Pi of a pass network, can be detected by a sequence of test vectors if at least one Pj exists such that Vi<>Vj.

Proof: Since no two product terms Pi and Pj are true simultaneously, the conditions of Theorem 3 are met. The following test vectors can be applied to detect the fault.

1) Apply an input vector such that Pj=1.

2) Apply an input vector such that Pi=1 and assign the value of Vi such that it is opposite to the value of Vj assigned in step 1.

If Xi is s-op, then the output will maintain the value of Vj; otherwise, the output will undergo a transition.

Q.E.D.

B. Regular Pass Networks

The silicon area occupied by an IC chip depends largely on its structural regularity, and therefore the design of regular pass networks plays an important role.

Definition 4: A disjoint pass network is said to be regular if the intersection of the set of control variables and the set of pass variables is a null set.

In regular networks, the control lines can be routed vertically and the pass lines can be routed horizontally. The test set for these networks is complete.

Corollary 4: All faults except s-on control variable faults in a regular pass network are detectable.

Proof: Since a regular pass network is disjoint, by corollaries 1 and 3, all faults except s-on faults are detectable. By corollary 2, all s-on faults are detectable if logic assignments on the pass variables Vi and Vj can be varied. In a general network, some pass variables are constants 0,1; therefore, all s-on faults are not detectable.

Q.E.D.

The only limitation to detecting all s-on faults is the constraint that some pass terms are constants. Since detection of s-on faults requires that the pass terms be variable, a general solution to obtaining complete fault coverage is to provide a mechanism that will allow every pass term to be variable. This can be achieved by adding a test input that allows those pass terms that normally are constants to be variable for the purpose of applying test sequences.

Theorem 4: For test purposes, only those pass terms that generate a constant 1 need be controllable with test inputs.

Proof: To test for a s-on fault on Xi associated with Pi(Vi), it is necessary to enable some Pj(Vj) and to assign Vi so as to dominate Vj. A 0 dominates a 1. Therefore, Vi is assigned a 0, and Vj is assigned a 1. If under fault-free conditions Vi=0, then a s-on fault in Pi is testable. If an extra test input can control those cases where Vi=1 under fault-free conditions, such that Vi is forced to 0, then all s-on faults are detectable.

Q.E.D.

Theorem 4 states that regular structures are completely testable with the addition of a single test input signal. The test input signal can be applied through an inverter to the pass leads. When the test input is 0, the pass terms are 1; when the test input is 1, the pass terms are 0 and s-on faults are detectable. The network shown in Figure 4 is a regular pass network whose test set is complete.

VI. CONCLUSION

Necessary and sufficient conditions for test generation in MOS switch logic circuits have been presented. Not all circuits have been found to be completely testable. A special design procedure is presented that produces circuits that have complete fault coverage.

One of the assumptions in this presentation was that whenever a s-on fault enables two paths to present a conflicting pair of signals at the output (one path generating a logic 1 and another a logic 0), the resulting output was either defined when a pull-up device was present, or was undefined when this device was not present. If no pull-up devices are present, then the output of such a conflict is determined by the electrical characteristics of the circuit. If Vdd and Vss are on the inputs of two conducting paths, it would be reasonable to conclude that high current would occur, resulting in an open circuit. If it were true that the driving inputs to a switch logic network were such that the high current that results from a s-on fault would force an open circuit, then every s-on fault would become a s-op fault. In this case, the only fault conditions that need be tested would be s-op for both pass and control variables and stuck-at faults for the pass variables. This results in a reduced test set.

REFERENCES

[1] C. A. Mead and L. A. Conway, Introduction to VLSI Systems. Reading MA: Addison-Wesley, 1980.

[2] Y. M. El-Ziq and S. Y. H. Su, "Fault Diagnosis of MOS Combinational Networks," IEEE Trans. Comput., Vol. C-31, pp. 129-139, Feb. 1982.

[3] Z. Kohavi, Switching and Finite Automata Theory. New York: McGraw-Hill, 1978.

[4] J. P. Hayes, "A Fault Simulation Methodology for VLSI," Proc. of the Nineteenth Design Automation Conference, Las Vegas, June 1982.

[5] R. L. Wadsack, "Fault Modeling and Logic Simulation of CMOS and MOS Integrated Circuits," The Bell System Technical Journal, Vol. 57, pp. 1449-1474, May-June 1978.

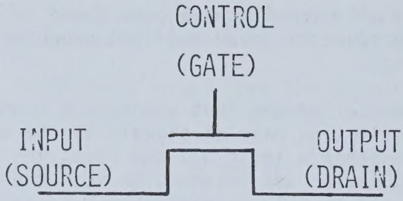
[6] Y. L. Levendel, P. R. Menon and C. E. Miller, "Accurate Logic Simulation Models for TTL Totempole and MOS gates and Tristate Devices," The Bell System Technical Journal, Vol. 60, pp. 1271-1287, Sept. 1981.

[7] J. Galiay, U. Crouzet and M. Vergniault, "Physical versus Logical Fault Models MOS LSI Circuits: Impact on Their Testability," IEEE Trans. Comput., Vol. C-29, pp. 527-531, June 1980.

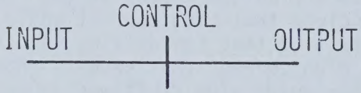
[8] J. Galiay, "Design of easily testable LSI circuits" (in French), Docteur de Specialite thesis, Univ. Paul Sabatier, Toulouse, France, Nov. 15, 1978, available from LAAS.

[9] D. Radhakrishnan, "Theory and Design of Pass Transistor Switching Circuits," Ph.D. dissertation, Dept. Elec. Engg., Univ. of Idaho, Moscow, May 1983.

[10] D. Radhakrishnan and G. K. Maki, "Design of Pass Transistor Switching Circuits," (submitted to IEEE Trans. Comput.)


CONTROL
(GATE)
INPUT
(SOURCE)
OUTPUT
(DRAIN)

ELECTRONIC REPRESENTATION


CONTROL
INPUT
OUTPUT

LOGIC REPRESENTATION

INPUT	CONTROL	OUTPUT
0	0	HIGH IMPEDANCE
0	1	0
1	0	HIGH IMPEDANCE
1	1	1

TRUTH TABLE

FIGURE 1. PASS TRANSISTOR REPRESENTATION

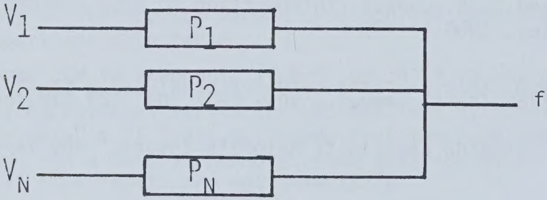

V_1
P_1
V_2
P_2
f
V_N
P_N

FIGURE 2. A GENERALIZED PASS NETWORK

$f = P_1(V_1) + P_2(V_2) + \ldots + P_N(V_N)$

CD \ AB	00	01	11	10
00	1	0	0	1
01	0	1	1	1
11	0	1	1	0
10	1	0	0	0

FIGURE 3. KARNAUGH MAP EXAMPLE

$f = A'B'(D') + AB'(C') + B(D)$

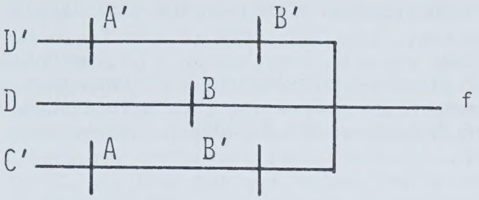


FIGURE 4. PASS NETWORK FOR THE FUNCTION:
$f = A'B'(D') + B(D) + AB'(C')$

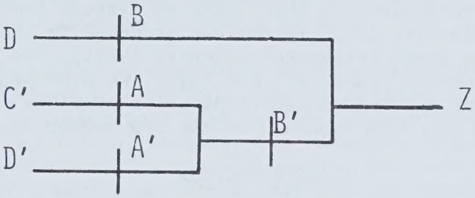


FIGURE 5. PASS NETWORK FOR THE FUNCTION:
$Z = B(D) + B'(A(C') + A'(D'))$

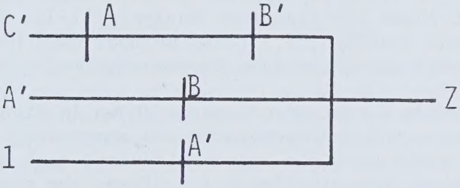


FIGURE 6. PASS NETWORK FOR THE FUNCTION:
$Z = AB'(C') + B(A') + A'(1)$

COMMENTS ON A DATA COMPRESSION TECHNIQUE FOR VLSI TESTING

KEWAL K. SALUJA
Department of Electrical & Computer Engineering
University of Newcastle, N.S.W. 2308
Australia.

R. DANDAPANI
Department of Mathematics
Youngstown State University
Youngstown, Ohio, 44555,U.S.A.

ABSTRACT

With the growth in complexity of logic in Integrated Circuits (ICs), considerable attention is being paid to the data compression techniques used for testing such ICs. In this paper we discuss some of the deficiencies of Built-in Testing (BIT) approaches proposed in the literature for testing Very Large Scale Integrated Circuits (VLSI). We give a characterization of functions for which BIT may fail. We also propose a remedy for this deficiency and outline an algorithm which provides better fault coverage. Advantages and disadvantages of the new scheme are also discussed.

1. INTRODUCTION

The growth of circuit complexity in VLSI chips has results in exponential growth in test complexity. Test generation is often difficult and test data size is much too large to use conventional testing methods. Therefore, a number of different approaches have been suggested to compress test data. Some of these are Signature Analysis [1-3], Syndrome Testing [4], Transition Count Testing [5]. These methods have been further classified as weighted and unweighted data compression methods in [6].

These methods have a test structure as shown in Figure 1 for single output circuits, where the test generator and compressor circuits may be on or off-chip. For the multiple output circuits, each output feeds a separate compressor as shown in Figure 2. Often, the compressors used are counters, normalized counters, or linear feedback Shift Registers (LFSRs). Recently, for multiple output circuits, a test structure as shown in Figure 3 was proposed in [7], which uses multiple input LFSR. This method led to the study of Built-in Testing (BIT) techniques. Some applications of BIT in VLSI have also been proposed in [8]. In this approach circuit on a chip is partitioned into subcircuits, with each subcircuit having its own Test Generator (TG) and Signature Analyzer (SA).

In Section 2 of this paper, we describe BIT and one of its generalizations called Store and Generate Testing (SGT) given in [9]. In Section 3 we show the limitations of these methods by way of an example. Section 4 describes notation and characterizes functions for which these methods may fail to detect faults. Using the characterization of Section 4 we demonstrate in Section 5, that BIT may fail to detect faults in a full adder and parallel adder part of an Arithmetic and Logic Unit (ALU). In Section 6 we propose an algorithm which increases the fault detection properties while using BIT and SGT Techniques. This algorithm is then compared with existing methods.

2. BUILT-IN TESTING (BIT)

A VLSI Circuit is partitioned into subcircuits called modules (M). Each M has its own TG and SA. Typically, each TG is a single input LFSR, and SA is a multiple input LFSR. Furthermore, the LFSR's are strung together to form a single-input/single-output shift register, as in LSSD [10] or equivalent methods [11]. In BIT, this shift register is used to initialize all TG's and SA's to a known starting state for testing. Exhaustive tests are then applied by TG's and the results are compressed by SA's. The final state of the SA's are called signatures of the modules. The signatures are read out and compared to fault-free circuit module signatures, stored separately, to detect faulty modules. Thus, TG and SA LFSR's can operate in TG, SA, Read Out and Normal System modes. Figure 4(a) and (b) shows an 8 input LFSR and associated multiplexer (this is a modification of the design proposed in [7,11]. In SA mode, the LFSR accepts 8 parallel inputs z_1, z_2, ..., z_8 as shown in Figure 4(c), the Read Out and Normal modes are shown in Figures 4(d) and 4(e) respectively. In TG mode, the LFSR acts like an independent LFSR with $z_1 = z_2 =$ $z_1 = z_2 = \ldots = z_8 = 0$ as shown in Figure 4(f).

In SGT, an M is tested by starting a TG in a known initial state and exercising M for a preset time period. The TG is reset to a new state and M is exercised for another preset time period. This process is repeated for all on-chip stored initial states. The resulting signature is compared with fault-free signature of M, stored on chip. The intermediate signature, resulting after each preset time period, may also be compared with its corresponding fault-free signature. Thus SGT is a generalization of BIT approach as a design alternative to the test hardware on the chip. In their paper, Agarwal and Cerny [9] do not identify any resulting test benefits. For example, it is unclear whether it tests more faults than BIT.

3. LIMITATIONS OF BIT AND SGT

In this section, we shall first demonstrate how BIT can fail to detect certain faults. For the sake of simplicity, we will consider a three bit TG and SA and three variable functions f_1 and f_2 as shown in Figure 5. Function f_3 can be any arbitrary function, but for the convenience of our analysis we shall assume $f_3 = 0$. Let us assume the initial states for both TG and SA to be 100. The TG will generate an input sequence as shown in Figure 6(a), the corresponding values of functions f_1 and f_2 are listed in Figure 6(b) and the states of the SA are listed in Figure 6(c). Now consider the faults α s-a-1 and β s-a-1. The faulty function values are listed in Figure 6(d) for the input sequence of Figure 6(a), and the states of SA under fault conditions are listed in Figure 6(e). The signature resulting from the fault-free and faulty modules are identical. They are both 100. Thus, BIT does not detect these faults. Even SGT may fail to detect these faults unless the state of the SA was read-out at the correct time. For example, the first and only time the states differ is when the input is 110, resulting in a fault-free state of SA being 111 and faulty state being 011. The SGT proposed in [9] does not guarantee this read-out unless all intermediate states are considered. This may necessitate a prohibitively large amount of extra hardware on the chip.

4. CHARACTERIZATION OF UNDETECTABLE FAULTS IN BIT

An examination of the example in the previous section shows that the fault α s-a-1 and β s-a-1 are individually detectable by BIT but become undetectable when both are present. Further, the effect of α s-a-1 propagates through the multiple input linear feedback shift register to cancel the effect of β s-a-1 due to the presence of EXCLUSIVE-OR gate. To characterize the untestability of faults in BIT, the following notations and definitions will be used:

Let M be the module under consideration with n inputs and m outputs. The input vectors to M are generated by TG and are denoted by LSFR_i,

$0 \leq i \leq 2^n-1$. Notice, we assume a TG generates all possible sequences of length N, therefore it may have some non-linearity. Let $f = (f_1, f_2, \ldots, f_m)$ be the output vector of M, where f_j, $1 \leq j \leq m$, is the j-th component connected to z_j input of the SA for M. Thus $f_j(LSFR_i)$ will denote the value of output f_j when the input to M is $LSFR_i$. We shall also denote this as $f_j(i)$ for notational simplicity. Also, in subsequent discussion we shall assume $m = n$.

The effect of faults will be indicated by superscripts. Thus f_j^σ indicates the output f_j in the presence of fault σ. Let $e_j^\sigma = f_j \oplus f_j^\sigma$ be the effect function for fault σ on function f_j.

Now consider the SA structure. Let Q_j be the j-th output of SA. We will say (Q_{j_1}, Q_{j_2}), $j_2 > j_1$, is a Feedback-free set (FFS) if the outputs $Q_{j_1}, Q_{j_1+1}, \ldots, Q_{j_2-1}$ are not fed-back in the SA structure. For example, in Figure 4(a), (Q_1, Q_3) is an FFS. The other FFS's in Figure 4(a) are (Q_4, Q_5) and (Q_6, Q_7).

Let $S^i = (s_1^i, s_2^i, \ldots, s_n^i)$ denote the state of SA after $LFSR_i$ has been applied to M. If each application of an input takes a unit time, S^i can be interpreted as the signature of M at i^{th} time instant from the initial state. $S^{i,\Sigma}$ will mean the state of SA at time i in the presence of the faults Σ in M.

Observation: $s_j^i = s_{j-1}^i \oplus f_j(i)$, $1 < j \leq n$.

Definition 1: $m_j(k)$ is a function f_j such that $f_j(LFSR_i) = 1$ if and only if $i = k$.

Result 1: In a module M if

(i) $f_j = m_j(k)$, $f_{j+1} = m_{j+1}(k+1)$

(ii) Q_j and Q_{j+1} are in an FFS

(iii) $\Sigma = \{f_j \text{ s-a-0 and } f_{j+1} \text{ s-a-0}\}$

then the fault set Σ is undetectable by BIT.

Proof: It is evident that $S^{k-1} = S^{k-1,\Sigma}$, because $f_j(\ell) = 0$ for all $\ell < k$. We can now write the states of SA for inputs $LFSR_k$ and $LFSR_{k+1}$ for fault-free and faulty conditions. As only the outputs f_j and f_{j+1} are affected by the fault, we need only consider the states s_j, s_{j+1}, s_{j+2} if Q_{j+1} is not being fed back and s_j, s_{j+1}, s_{j+2} and s_1 if Q_{j+1} is being fed back.

Case 1: Q_{j+1} is not being fed back

$$s_j^k = s_{j-1}^{k-1} \oplus f_j(k) = s_{j-1}^{k-1} \oplus 1 \quad (1)$$

$$s_{j+1}^k = s_j^{k-1} \oplus f_{j+1}(k) = s_j^{k-1} \quad (2)$$

$$s_{j+2}^k = s_{j+1}^{k-1} \oplus f_{j+2}(k) \quad (3)$$

$$s_j^{k+1} = s_{j-1}^k \oplus f_j(k+1) = s_{j-1}^k \tag{4}$$

$$s_{j+1}^{k+1} = s_j^k \oplus f_{j+1}(k+1) = s_{j-1}^{k-1} \tag{5}$$

$$s_{j+2}^{k+1} = s_{j+1}^k \oplus f_{j+2}(k+1) = s_j^{k-1} \oplus f_{j+2}(k+1) \tag{6}$$

However, in the presence of fault Σ the equations (1) to (6) become

$$s_j^{k,\Sigma} = s_{j-1}^{k-1} \tag{7}$$

$$s_{j+1}^{k,\Sigma} = s_j^{k-1} \tag{8}$$

$$s_{j+2}^{k,\Sigma} = s_{j+1}^{k-1} \oplus f_{j+2}(k) \tag{9}$$

$$s_{j+1}^{k+1,\Sigma} = s_{j-1}^k \tag{10}$$

$$s_{j+1}^{k+1,\Sigma} = s_j^{k,\Sigma} = s_{j-1}^{k-1} \tag{11}$$

$$s_{j+2}^{k+1,\Sigma} = s_{j+1}^{k,\Sigma} \oplus f_{j+2}(k+1) = s_j^{k-1} \oplus f_{j+2}(k+1) \tag{12}$$

From equations (4), (5), (6) and (10), (11), (12) we conclude

$$s_j^{k+1,\Sigma} = s_j^{k+1} \; ; \; s_{j+1}^{k+1,\Sigma} = s_{j+1}^{k+1} \text{ and } s_{j+2}^{k+1,\Sigma} = s_{j+2}^{k+1}$$

Therefore $S^{k+1} = S^{k+1,\Sigma}$.

It is now easy to see that $S^{2^n} = S^{2^n,\Sigma}$.

Thus, the final signatures in fault-free and faulty cases will be identical. Therefore, Σ is undetectable.

Case 2: Q_{j+1} is being fed back.
In this case we also need to consider s_1^{k+1}. It is a simple matter to check that $s_1^{k+1,\Sigma} = s_1^{k+1}$ for fault set Σ. ∇∇∇

Result 2: A fault set Σ is undetectable by BIT if

(i) Σ affects only f_j and f_{j+1} i.e. $f_\ell = f_\ell^\Sigma$ for all $\ell \neq j$ and $\ell \neq j+1$

(ii) the effect functions $e_j^\Sigma = m_j(k)$ and $e_{j+1}^\Sigma = m_{j+1}(k+1)$

(iii) Q_j and Q_{j+1} are in an FFS.

Proof: The proof is similar to the proof of Result 1. For example, we can conclude $S^{k-1} = S^{k-1,\Sigma}$ because $f_j(\ell) = f_j^\Sigma(\ell)$ for all $\ell < k$ and $j = 1, \ldots, n$. Using arguments as in Result 1, we can show that $S^{k+1} = S^{k+1,\Sigma}$ and therefore, $S^{2n} = S^{2n,\Sigma}$. ∇∇∇

The effect functions of Result 2 can be generalized, which leads to Result 3. This is stated without proof as it is conceptually simple but notationally complex.

Result 3: A fault Σ is undetectable by BIT if

(i) as in Result 2

(ii) $e_j^{\Sigma}(k) = e_{j+1}^{\Sigma}(k+1)$ for all k

(iii) as in Result 2

The FFS used in Result 2 is limited to two consecutive adjacent outputs of M. However, if f_{j_1} and f_{j_2} are such that $Q_{j_1}, Q_{j_1+1}, \ldots, Q_{j_2}$ are all contained in one FFS, then Result 2 can be generalized as follows:

Result 4: A fault set Σ is undetectable by BIT if

(i) Σ affects only f_{j_1} and f_{j_2}

(ii) $e_{j_1}^{\Sigma}(k) = e_{j_2}^{\Sigma}(k+j_2-j_1)$ for all k

(iii) $Q_{j_1}, Q_{j_1+1}, \ldots, Q_{j_2}$ are all in one FFS

5. EXAMPLES

In this section, we present two examples. In the first example, we demonstrate that a single fault may also be undetectable by BIT. Once again for simplicity we consider the case of n = 3.

Example 1: Consider the realizations of functions $f_1 = x_1\, \bar{x}_2\, \bar{x}_3 + x_1\, x_2\, x_3$ and $f_2 = x_1\, \bar{x}_2\, x_3 + \bar{x}_1\, x_2\, x_3$ as shown in Figure 7. Now with fault α s-a-0, we have $f_1^{\alpha} = x_1\, \bar{x}_2\, \bar{x}_3$ and $f_2^{\alpha} = x_1\, \bar{x}_2\, x_3$. Thus $e_1^{\alpha} = x_1\, x_2\, x_3$ and $e_2^{\alpha} = \bar{x}\, x_2\, x_3$. It is easy to see that for our TG sequence $e_1^{\alpha} = m_1(k)$ and $e_2^{\alpha} = m_2(k+1)$ for k = 111. Thus, by Result 2 this fault is undetectable by BIT.

Example 2: Consider a realization of a full adder as follows:

$$S_i = \bar{a}_i\, \bar{b}_i\, c_i + \bar{a}_i\, b_i\, \bar{c}_i + a_i\, \bar{b}_i\, \bar{c}_i + a_i\, b_i\, c_i$$

$$C_{i+1} = a_i\, b_i + b_i\, c_i + a_i\, c_i$$

In a carry save adder S_i and C_{i+1} may feed two storage elements and the full adder may derive its inputs a_i, b_i, c_i from storage elements, thus forming a well suited structure for application of BIT. Now consider stack type faults α, β such that

$$S_i^{\alpha} = \bar{a}_i\, \bar{b}_i\, c_i + \bar{a}_i\, b_i\, \bar{c}_i + a_i\, b_i\, c_i$$

$$C_{i+1}^{\beta} = a_i\, b_i + b_i \cdot 1 + a_i\, c_i = b_i + a_i\, c_i$$

It is now easy to check that the conditions for undetectability of these faults, as given in Result 2, are satisfied, thus making these faults undetectable by BIT.

6. MULTIPLE SIGNATURE APPROACH TO TESTING

We have shown in Section 3 that the faults α and β in the circuits of Figure 5 are undetectable if the signature is read out at the completion of testing. However, it is also evident from the fault free and faulty SA states listed in Figures 6(c) and 6(e) respectively, that these faults are detectable if one reads the SA after the input 111. Thus, to detect faults which have been characterized in Section 4, one only need to read out the appropriate intermediate signatures. This may warrant simulation of certain faults and then choosing appropriate readout points. If, along with BIT, an underlying scan structure is also assumed then, while reading out the signatures, new initial values for TG's and SA's can be scanned in. The following algorithm summarizes this approach.

```
Algorithm: Step 1: set $k=0$.
           Step 2: Scan in the $k^{th}$ initializing pattern for TG's and SA's.
           Step 3: Take the circuit into BIT mode for a preset time and
                   collect signatures.
           Step 4: set $k = k+1$.
           Step 5: Read out SA's while scanning in $k^{th}$ initializing pattern.
           Step 6: Compare the signatures. If there is a mismatch, then
                   the circuit under test is faulty, need no further tests.
           Step 7: If the testing is complete -- stop testing else go to
                   Step 3.
```

The above scheme uses data compression as in BIT and also has all the advantages of scan schemes. The off-chip storage of initializing sequences for TG's and SA's will result in saving in silicon area. Judicious use of readouts will result in improved fault coverage. With this scheme, it is possible to use methods proposed in [12] for constrained optimization of time for testing the complete circuit.

The major disadvantage of the scheme being multiple readouts require longer time, more comparisons and increased test data size. As compared to conventional testing, the performance can be expected to be superior in all these respects.

7. CONCLUSIONS

Data compression techniques used for testing Integrated Circuits are analyzed. An example circuit with faults is presented which may not be completely tested by these methods. Using this example, a characterization of functions is developed for which Built-in Testing will fail to detect all faults. Examples are given, using the characterization developed by us for which BIT may fail to detect faults. One such example being a carry save adder. An algorithm is presented which increases fault coverage of existing Built-in Testing methods.

REFERENCES

[1] R. A. Frohwerk, "Signature Analysis: A new Digital Field Service Method", *Hewlett-Packard Journal*, May 1977, pp. 2-8.

[2] J. E. Smith, "Measures of the Effectiveness of Fault Signature Analysis", *IEEE Trans. Comp.*, Vol. C29, June 1980, pp. 510-514.

[3] D. K. Bhavsar and R, W, Heckelman, "Self Testing by Polynomial Division", *International Test Conference 1981*, Philadelphia, pp.208-216.

[4] J. Savir, "Syndrome-Testable Design of Combinational Circuits", *IEEE Trans. Comp.*, Vol. C-29, June 1980, pp. 442-451.

[5] J. P. Hayes, "Transition Count Testing of Combinational Logic Circuits", *IEEE Trans. Comp.*, Vol. C-25, June 1976, pp. 613-620.

[6] K.K. Saluja and C. S. Venkatraman, "Generalized Data Compression for VLSI Testing", *Proceedings of the Twentieth Allerton Conference on Communications, Control and Computing*, Illinois, October 1982, pp. 229-230.

[7] B. Konemann, J. Mucha and G. Zwiehoff, "Built-In Logic Block Observation Techniques", *Digest 1979 Test Conf.*, October 1979, pp. 37-41.

[8] *Proceedings of the International Test Conference, 1982*, Philadelphia.

[9] V. K. Agarwal and E. Cerny, "Store and Generate Built-In Testing Approach", *Proc. 11th Int. Symp. on Fault Tolerant Computing*, June 1980, pp. 35-40.

[10] E. B. Eichelberger and T. W. Williams, "A Logic Design Structure for LSI Testability", *Proc. 14th Design Automation Conf.*, June 1977, pp. 462-468.

[11] T. W. Williams and K. P. Parker, "Design for Testability - A Survey", *IEEE Trans. Comp.*, Vol. C-31, No. 1, January 1982, pp. 2-15.

[12] C. R. Kime and K. K. Saluja, "Test Scheduling in Testable VLSI Circuits", *Proc. 12th Int. Symp. on Fault Tolerant Computing*, June 1982, pp. 406-412.

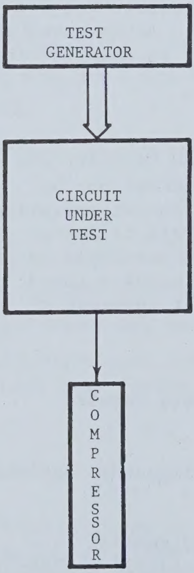

TEST GENERATOR
CIRCUIT UNDER TEST
COMPRESSOR

FIGURE 1: STRUCTURE FOR TESTING A SINGLE OUTPUT CIRCUIT

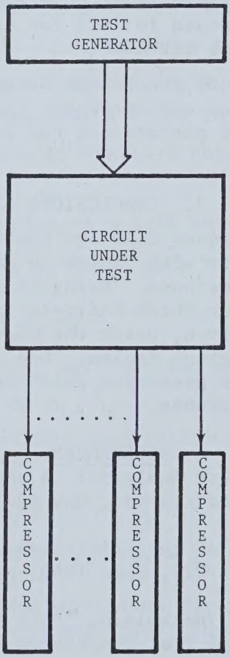

TEST GENERATOR
CIRCUIT UNDER TEST
COMPRESSOR
COMPRESSOR
COMPRESSOR

FIGURE 2: STRUCTURE FOR TESTING A MULTI-OUTPUT CIRCUIT

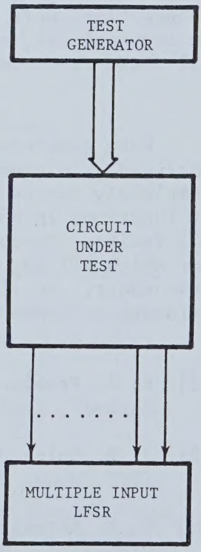

TEST GENERATOR
CIRCUIT UNDER TEST
MULTIPLE INPUT LFSR

FIGURE 3: STRUCTURE USING MULTIPLE INPUT LFSR

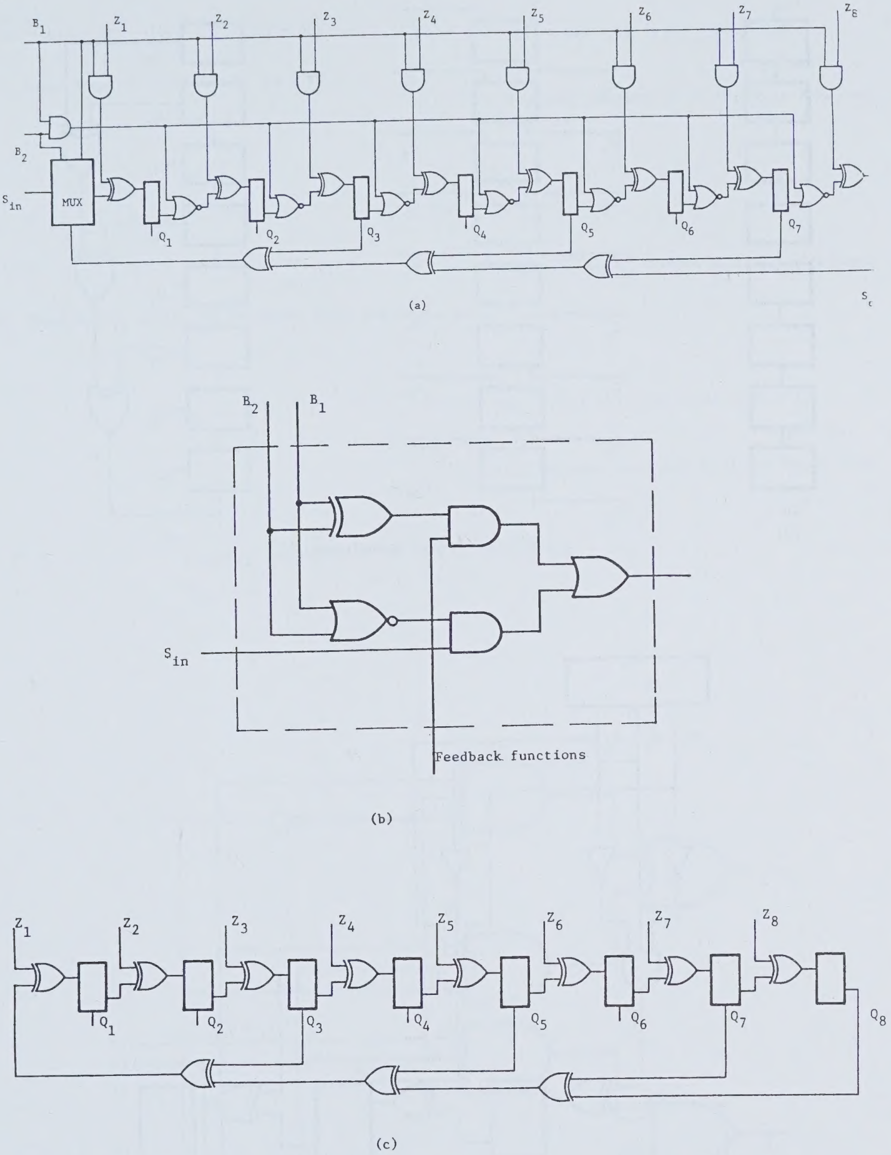
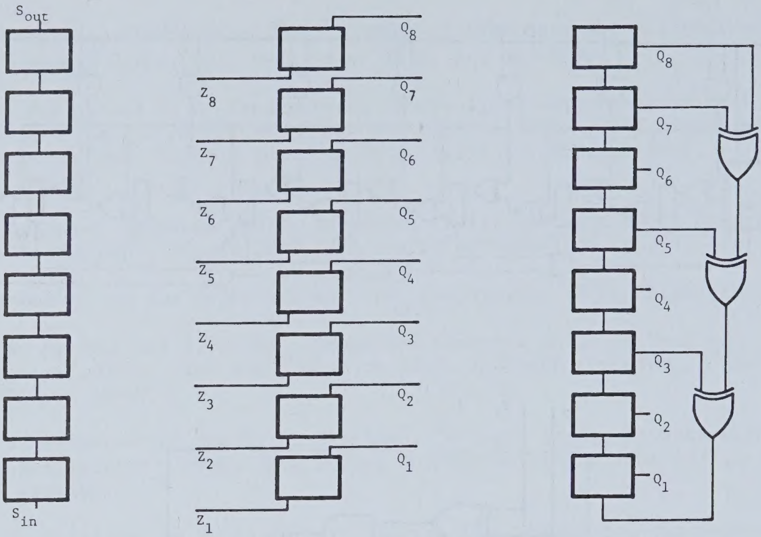

B1
B2
Z1
Z2
Z3
Z4
Z5
Z6
Z7
Z8
S_in
MUX
Q1
Q2
Q3
Q4
Q5
Q6
Q7
S_c
(a)
B2
B1
S_in
Feedback functions
(b)
Z1
Z2
Z3
Z4
Z5
Z6
Z7
Z8
Q1
Q2
Q3
Q4
Q5
Q6
Q7
Q8
(c)

FIGURE 4: A multiple input LRSR in different modes: (a) General form of LFSR; (b) Multiplexer; (c) SA mode: $B_1B_2 = 10$; (d) Read Out mode: $B_1B_2 = 00$; (e) Normal System mode: $B_1B_2 = 11$; (f) TG mode: $B_1B_2 = 01$.


S_out
S_in
Q8
Z8
Q7
Z7
Q6
Z6
Q5
Z5
Q4
Z4
Q3
Z3
Q2
Z2
Q1
Z1
Q8
Q7
Q6
Q5
Q4
Q3
Q2
Q1

(d) (e) (f)

FIGURE 4: (continued)

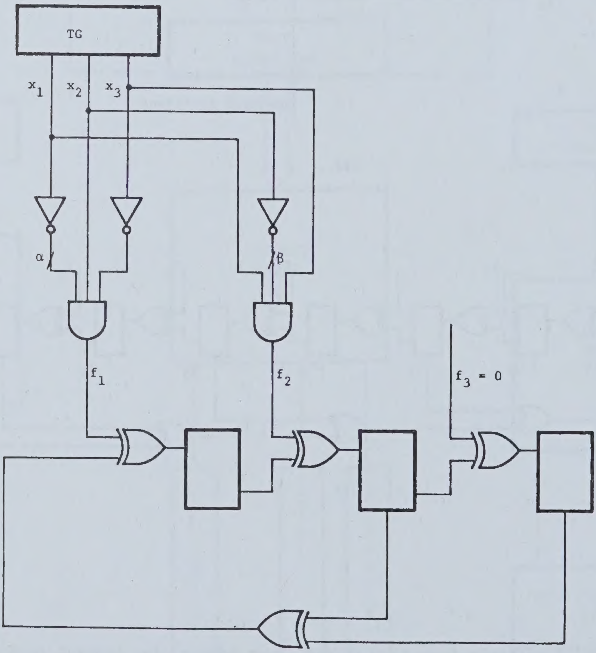

TG
x1
x2
x3
α
β
f1
f2
f3 = 0

FIGURE 5: A BIT REALIZATION OF EXAMPLE FUNCTIONS

$x_1x_2x_3$

100 → 010 → 101 → 110 → 111 → 011 → 001

(a) Input Sequence Generated By 3 Bit TG

f_1: 0 1 0 0 0 0 0

f_2: 0 0 1 0 0 0 0

(b) Functions f_1 and f_2 For The Above Input Sequence

100 → 010 → 001 → 110 → 111 → 011 → 001 → $\underbrace{100}_{\substack{\uparrow \\ \text{signature}}}$

(c) State Sequence Generated By SA For Fault Free Case

f_1^{α}: 0 1 0 1 0 0 0

f_2^{β} 0 0 1 0 1 0 0

(d) Function f_1^{α} And f_2^{β} For The Above Input Sequence

100 → 010 → 001 → 110 → 011 → 011 → 001 → $\underbrace{100}_{\substack{\uparrow \\ \text{signature}}}$

(e) State Sequence Generated By SA For Faulty Case

Figure 6: ANALYSIS OF THE CIRCUIT OF FIGURE 5

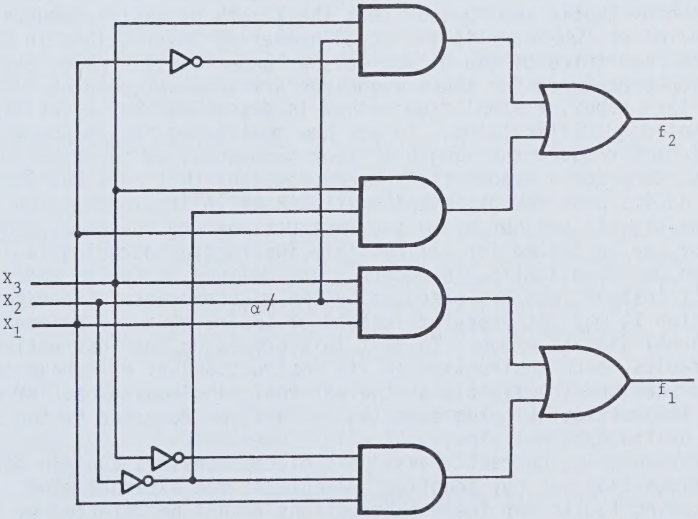


FIGURE 7: EXAMPLE OF AN UNTESTABLE SINGLE FAULT

A SIMPLIFIED METHOD FOR TESTING MICROPROCESSORS

HARVINDER SINGH
Electrical Engineering Department
University of Nevada-Reno
Reno, Nevada

JON G. BREDESON
School of Electrical Engineering and Computer Science
University of Oklahoma
Norman, Oklahoma

ABSTRACT

A Simplified method for testing microprocessors is presented. It employs monitoring the signals put out by the microprocessor at the external pins.

I. INTRODUCTION

The classical fault detection techniques based on the gate level or state level description of circuits are not suitable for testing Very Large Scale (VLSI) chips such as microprocessors. For these chips, functional level techniques hold a greater promise. A very important approach for functional testing of microprocessors has been described by Thatte and Abraham in their recently published paper 1. They have divided the total set of microprocessor faults into five subsets according to the functional category in which these faults occur. These fault categories are termed as instruction decoding faults, register decoding faults, data transfer faults, data storage faults and data manipulation faults. Test methods have then been described to test faults in each category.

The approach suggested by Thatte and Abraham is an excellent way of testing microprocessors with varied architectures. However, the test sequences to detect instruction decoding faults can be very lengthy. It has been shown by Thatte and Abraham that the length of test sequences is of the order of n_I^2 (where n_I is the total number of instructions in the instruction repertoire of the microprocessor under test). Also, the test generation procedures for these sequences are somewhat complex.

In this paper, a simplified method is described that alleviates the above mentioned difficulties. To see how monitoring the response at the external pins reduces the length of test sequences, we take the following example. Consider a hypothetical microprocessor that puts out different signals at its pins for different instructions in its instruction set; such that, the signals put out by no two instructions are the same. This microprocessor can be tested for all possible instruction decoding faults (instruction decoding faults, in general, are defined as faults under which incorrect instructions are executed by the microprocessor under test, i.e., instruction I_1 may get executed instead of instruction I_2) by monitoring the external signals alone. To test this processor for instruction decoding faults, each instruction in its instruction set will have to be executed once and the signals at the external pins monitored. If an incorrect instruction is being executed, it will be detected by the signals put out at the external pins.

Unfortunately, currently available microprocessors contain sets of instructions that put out identical signals at the external pins. Instruction decoding faults for these instructions cannot be detected by the above described method. However, the testing can still be simplified. To do this, we divide the total instruction set into subsets of instructions that put out identical signals at the output pins of the microprocessor. The instruction decoding faults within each of the subsets may now be detected using methods of Thatte and Abraham. If there are k such subsets of instructions containing $\eta_1, \eta_2, \ldots, \eta_k^2$ instructions each, then it can be shown the order of the least sequence to detect faults in these subsets is given by:

$$\Theta(\eta_I) = \eta_1^2 + \eta_2^2 + \ldots + \eta_k$$

where $\Theta(\eta_I)$ = order of sequences to detect instruction decoding faults.

Clearly, this results in a shorter test sequence since $\Theta(\eta_I) \leq \Theta(\eta_I^2)$

Also recommended in this paper is the introduction of two new instructions in currently manufactured microprocessors. The two new instructions are termed as MOVE MULTIPLE instructions and provide a means for simplifying the data transfer between various microprocessor registers and the outside world. It can be shown that if these new instructions are implemented on an Intel 8085 microprocessor and the testing is carried out while monitoring the signals at the external pins, the test time for instruction decoding faults is reduced significantly.

II. FAULT MODEL

Before we describe complete testing procedures used in testing the microprocessors, it is necessary to develop a fault model for microprocessors. The goal is to provide a sound theoretical footing for our fault detection techniques. Further, if the fault model detection methods are to be useful to microprocessor users, the fault model should be applicable to various commercially available microprocessors and yet be independent of their implementation details. The design of microprocessors does not easily lend itself to any theoretical analysis. The features found in commercially available microprocessors have evolved to fulfill the needs of the applications for which microprocessors are used. This makes the task of developing a comprehensive fault model very complicated. It is clear that generalizations and assumptions would have to be made. Two types of assumptions are necessary to develop the fault model: 1) assumptions regarding the architectures of the microprocessors, and 2) assumptions regarding the types of faults to be allowed in these microprocessors. The assumptions should fit the real world as closely as possible and should still simplify the testing.

2.1 Assumptions and Definitions for Developing the Fault Model

Assumption 2.1: To execute a valid instruction, every microprocessor must put out signals at its external pins.

Assumption 2.2: Any detectable fault within a microprocessor must either result in incorrect data being stored in various microprocessor registers (including registers like program counter, stack pointer, etc.) or must result in incorrect signals being output at various external microprocessor pins.

Assumption 2.3: If due to a fault in the instruction decoder of a microprocessor an incorrect instruction gets executed, and if the incorrect instruction that is executed:

- (i) has different instruction timing, or
- (ii) has different number of bytes, or
- (iii) has same number of bytes and execution time, but produces different signals at the external pins,

all the work registers A through L, the stack pointer (SP), the program counter (PC), status register (SR) and the instruction register (IR). The contents of the memory location pointed to by the address byte of MVMI are transferred to Register A (accumulator). The contents transferred into Register B are from a memory location having an address LOC + 1 (LOC = address stored in the address byte of the instruction MVMI). Contents transferred in registers C, D, E, H, L, SP, PC, SR and IR are from locations with addresses LOC + 2, LOC + 3, LOC + 5, ..., LOC + 12.

OP Code	Operand
MVMO	dbl. addr.

0 0 0 0 1 0 0 0
low-order addr.
high-order addr.

Cycles: 16
States: 49
Addressing: Indirect
Flags: None

It is anticipated that the execution of the above instruction on Intel 8085 will require 49 clock periods. This is arrived at using the following considerations. In Intel 8085, every machine cycle involving memory reference must take 3 clock periods, except for the instruction fetch cycle. Instruction fetch takes 4 clock periods to complete. Since MVMI requires 16 memory reference operations, it can be executed in 4 + 15 x 3 = 49 clock periods. It is also clear that this instruction will involve 16 machine cycles, one for each memory reference operation.

MVMO (MOVE MULTIPLE OUT)

This instruction performs the function opposite to that of MVMI. It transfers the contents of various microprocessor registers into consecutive memory locations in the external memory. The microprocessor registers involved are: al the microprocessor registers A through E plus the status register. The contents of the accumulator A are transferred to the memory location addressed by the address byte of MVMO. Contents of registers B, C, D, E, H, L, PC, SR and IR are transferred to locations LOC, LOC + 1, ..., LOC + 12. (LOC is the address stored in the address bytes of the MVMO instruction.)

OP Code	Operand
MVMO	dbl. addr.

0 0 0 0 1 0 1 0
low-order addr.
high-order addr.

Cycles: 16

then this fault can be detected solely by monitoring the response at the external pins (i.e. the incorrect data stored in the various registers need not be examined).

In order to facilitate the presentation of the remainder of the theory, we introduce two new definitions.

Definition 2.1: External Pin Distinguishable (EPD) Instructions: If the signals produced at the external microprocessor pins in response to the execution of two instructions I_1 and I_2 are different regardless of data, I_1 and I_2 are said to be external pin distinguishable.

Definition 2.2: External Pin Indistinguishable (EPI) Instructions: If the signals produced at the external microprocessor pins in response to execution of two instructions I_1 and I_2 are the same, regardless of data, I_1 and I_2 are said to be external pin indistinguishable.

Assumption 2.4: Only the following types of instruction decoding faults are allowed:

(i) $f(I_i/\phi)$

(ii) $f(I_i/I_k)$. I_j and I_k can be either EPD or EPI instructions.

(iii) $f(I_i/I_i + I_j)$, $f(I_i/I_i + I_j + I_k)$, $f(I_i/I_i + I_j + I_k + \ldots I_n)$ etc. Instructions $I_j, I_k, \ldots, I_n$ do not belong to the same group of EPI instructions to which I_i belongs.

(iv) Any other faults in the instruction decoding or control sequence that can alter the microprocessor signals put out at the external pins.

(Note that $f(I_1/I_2)$ denotes an instruction decoding fault that causes the instruction I_2 to be incorrectly decoded instead of I_1. Similarly, $f(I_1/I_2 + I_3)$ denotes a fault that incorrectly causes the execution of I_2 and I_3 instead of I_1, etc. For a complete detailed description and classification of faults as used here, please refer to (1) or (2).

The above definitions and assumptions define the fault model that will be used later for describing the test method. However, before describing the test method, we also briefly discuss two new instructions that can be easily incorporated into microprocessors for simplifying the testing.

III. MOVE MULTIPLE INSTRUCTIONS AND THEIR APPLICATION TO SIMPLIFY THE TESTING OF MICROPROCESSORS

Most currently manufactured microprocessors do not use all the possible available bit patterns in the instruction register to implement the instructions. For example, consider the Intel 8085. The bit patterns corresponding to the following hexadecimal numbers are unused: 08, 10, 18, 28, 38, CB, D9, DD, ED, FD. Two of these bit patterns 08 and 10 can be used to implement the instructions whose description is given below.

MVMI (MOVE MULTIPLE IN)

This is a three byte instruction that inputs the contents of the consecutive memory locations of the external memory into the various microprocessor registers. The registers involved are:

States: 49
Addressing: Indirect
Flags: None

As in the case of the instruction MVMI, we anticipate that MVMO will also require 49 clock periods. This is a reasonable execution time to assume since the instruction requires 16 memory reference operations. Also, we assume that the instruction has 16 machine cycles, one cycle for each memory reference operation.

3.1 Use of the 'MOVE MULTIPLE' Instructions to Detect Data Storage and Data Transfer Faults

Since data storage faults refer to the stuck-at faults within the microprocessor registers, these faults can be easily detected by writing in and reading out various data in the microprocessor registers. Similarly, data transfer faults can be detected by transmitting different values of data over the microprocessor bus. It follows that if the microprocessor has only one bus then the faults in that bus can be detected by transmitting all possible values of data over that bus. Also, the faults in the various microprocessor registers can be detected by writing in and reading out all the possible values of data in each of these registers. Therefore, the following test procedure tests all the Data Transfer and Data Storage faults.

Test Algorithm 3.1.1

```
1. Let K = 0
2. MVMI LOC 1 /Execute this instruction with the binary value
               of K stored in each of the memory locations
               LOC 1 through LOC L + 12/
3. MVMO LOC 2
4. Increment K
5. If K = 2 , Stop
6. If K < 2 , Go to 1.
```

Since the procedure above reads in and writes out every possible bit pattern in each register of the microprocessor, it is obvious that it tests all the data storage faults. Also, since all the possible bit patterns are also transmitted over the bus, the procedure also tests all the data transfer faults. It is assumed that the 'MOVE MULTIPLE' instructions that appear in the above test procedure have been checked and found out to be fault free.

3.2 Use of the 'MOVE MULTIPLE' Instructions to Detect Instruction Decoding Faults

In general, any fault of the type $f(I_i/I_j)$ can be detected by executing the following test sequence and monitoring the signals produced at the external microprocessor pins.

Test Sequence 3.2.1

```
1. MVMI LOC /Memory locations LOC through LOC + 12 contain data
             that is chosen using the methods explained in
             (2).
2. $I_1$
3. MVMO LOC
```

It is assumed that data input by the above MVMI instruction can be chosen such that no two instructions operating on this data produce identical results in the microprocessor registers. It is easy to see why the above procedure detects all faults of the type $f(I_i/I_j)$. If instead of instruction I_i any other instruction I_j is executed in Step 2, it will produce different results in microprocessor registers than those that would be produced by I_i. This data is output in Step 3 of the above sequence by instruction MVMO and stored in memory location LOC 1 through LOC 1 + 12. When this data is compared to the reference expected output data, any fault of the type $f(I_i/I_j)$ would be detected.

The two procedures described above reduce the test time considerably for data transfer, data storage and instruction decoding faults. The data manipulation faults (i.e. the faults in the ALU, etc.) may be tested by the methods described in (1). The methods described herein can be used in conjunction with some methods described in (1) to completely test the microprocessor.

IV. CONCLUDING REMARKS

It is also felt that the test method suggested here is an improvement over the methods of Thatte and Abraham in at least three ways: (a) first, the tests are easier to generate, (b) second, the fault model is less restricted and covers a larger number of faults, (c) third, the testing time is considerably reduced.

(a) Ease of Test Generation. The test method given here is simpler to use than the methods of Thatte and Abraham for the detection of Data Transfer, Data Storage or Instruction Decoding Faults.

(b) Fault Model. The fault model assumed by Thatte and Abraham is very restricted since it allows for the existence of only those faults that are caused by a single stuck-at fault in the instruction decoder. However, the fault model presented here allows for the detection of all those faults that either alter the signal's output at the external microprocessor pins or cause incorrect data to be stored in the microprocessor registers.

(c) Testing Time. The testing time for microprocessors is considerably reduced by using the method described here. This is so because the present method considerably reduces the length of the test sequences to detect the instruction decoding faults, which form the major portion of the total test sequence.

V. REFERENCES

1. Satish M. Thatte and Jacob A. Abraham, "Test Generation for Microprocessors," IEEE Transactions on Computers, Vol. C-29, No. 6, June 1980.

2. H. Singh, "Design Modifications in Microprocessors to Simplify Their Testing," Ph.D. Dissertation, University of Oklahoma, Norman, Oklahoma, December 1982.

PLA IMPLEMENTATION OF K-OUT-OF-N CODE TSC CHECKER*

BELLA BOSE and DER JEI LIN
Department of Computer Science
Oregon State University
Corvallis, Oregon 97331

ABSTRACT

PLA implementation of totally self checking (TSC) checkers for k-out-of-n codes, $2 \le k \le n-2$, is presented. For k-out-of-2k, k-out-of-2k+1, k+1-out-of-2k+1, and k±1-out-of-2k codes, TSC checkers are designed using only one PLA. TSC checkers for all other codes are designed using 2 PLAs, and in fact the second PLA is very small for most of the codes.

I. INTRODUCTION

In designing a highly reliable system, using self checking checkers is very advantageous, since; first, tampering of data is avoided because errors are detected no sooner than they occur; second, faults both transient and permanent can be detected during the normal operation; and third, necessary software diagnostic programs are made much simpler or even eliminated. In order to have a high throughput rate the checking process must be as fast as possible and hence the checker must have the least possible logic levels.

The concept of self checking circuit was first introduced by Carter and Schneider [1] and formulated by Anderson and Metze[2] and Wakerly [3]. In [1], Anderson and Metze have given a procedure for the design of totally self checking (TSC) checkers for k-out-of-2k codes, which we refer as k/2k codes. Any k/n code checker is designed by first converting the given code to a $1/\binom{n}{k}$ code which is again converted into an appropriate $k'/2k'$ code using a self checking translator. Here k' must satisfy the condition, $2^{k'} \le \binom{n}{k} \le \binom{2k'}{k'}$. Whenever $\binom{n}{k}=3$ or 7 no k' satisfies this condition. However, solutions for these two cases are given by Reddy [4] and David [5]. In [4], the two level TSC checkers for k/2k+1 and k+1/2k+1 codes are given. Many properties of TSC checkers are discussed by Smith and Metze in [6]. Marouf and Friedman [7] have developed efficient multi-level TSC checkers for k/n codes. Using their method, PLA realization of TSC checkers for a subclass of k/n codes is given by Wang and Avizienis [8]. Further results on TSC checkers for constant weight codes are given in [9-13].

In this paper we have designed TSC checkers for k/n codes, where $2 \le k \le n-2$, using at most 2 PLAs. The design procedure given here, first converts the given k/n code to a 1/m code, where m satisfies the following condition:

$$\min\,[n-k,\ k] \ge \left\lceil \frac{n}{m} \right\rceil \qquad (1\text{-}1).$$

This conversion can be done using a 2-level AND-OR logic circuit and this circuit satisfies the TSC checker properties. When n=2k (i.e. k/2k codes), the value of m turns out to be 2 (i.e. output is a 1/2 code). This gives the Anderson-Metze TSC checker. For other codes, the value of m will be in the range $2 \le m \le \left\lceil \frac{n}{2} \right\rceil$. In fact, for most of the practical cases the value of m is very small. For example, when $4k \ge n \ge 2k$, the value of m will be less than or equal to 4. Thus, for these codes, a TSC checker with output code space 1/m, where $m \le 4$, can be designed using 2-level AND-OR circuit.

To get a k/n to 1/2 code TSC checker, the 1/m code is converted to an appropriate $k'/2k'$ code using a self checking translator, similar to the one described in [2]. According to the design rule given in [2], k' must satisfy the condition

$$2^{k'} \leq m \leq \binom{2k'}{k'} \qquad (1\text{-}2).$$

The self checking translator is nothing but a series of single level $2k'$ OR gates and thus these gates can be merged with the previous level OR gates. By this way we can get a k/n code to $k'/2k'$ code TSC checker using only one PLA. Another PLA, which is a $k'/2k'$ code to 1/2 code TSC checker, can be cascaded to the first PLA. By this way a TSC checker for any k/n code can be designed using at most 2 PLAs. The size of the second PLA is very small for most of the codes. For example, when $2k < n \leq 4k$, the second PLA is nothing but a TSC checker circuit with input space 2/4 code and output space 1/2 code.

We have also designed TSC checkers for k/2k+1 and k+1/2k+1 codes using only one PLA and this gives a TSC checker equivalent to the one given by Reddy [4]. Moreover using only one PLA, the TSC checkers for k-1/2k and k+1/2k codes are given. So far no 2-level TSC checker is known for k±1/2k codes.

In Section II, the properties of a TSC checker are discussed. The design procedure for any k/n code TSC checker is developed in Section III. By slightly modifying the design procedure developed in Section III, the 2-level TSC checkers are developed for k/2k+1 and k+1/2k+1 codes in Section IV and k±1/2k codes in Section V.

II. PROPERTIES OF A TSC CHECKER

Let I, $\overline{I}$, A, and $\overline{A}$ represent the set of input code space, input non-code space, output code space and output non-code space respectively. The output function of a circuit is represented by Z. Let F represent the set of faults in a circuit.

Definition: A circuit is self testing for the fault set F if for every fault $f \in F$, the circuit produces a non-code space output for at least one code space input.

i.e. $\forall\ f \in F,\ \exists\ a \in I$ s.t. $Z(a,f) \in \overline{A}$.

Definition: A circuit is fault secure for the fault set F, if, for every fault $f \in F$, the circuit never produces an incorrect code space output for any code space input.

i.e. $\forall\ f \in F,\ \forall\ a \in I$ either $Z(a,f) = Z(a)$
or $Z(a,f) \in \overline{A}$.

Definition: A circuit is totally self checking for the fault set F if it is both self testing and fault secure for the fault set F.

Definition: A circuit is code disjoint if it maps the input code space to the output code space and the input non-code space to the output non-code space.

i.e. $\forall\ a \in I,\ Z(a) \in A$ and $\forall\ b \in \overline{I}\ \ Z(b) \in \overline{A}$.

Definition: A circuit, which is both totally self checking and code disjoint is called a totally self checking (TSC) checker.

Remark: Notice that in the definition of 'code disjoint' the condition is described under the assumption of faultless mapping. Thus, in the definition of TSC, the three conditions are compatible.

The self testing property ensures that all faults produce detectable errors whereas the fault secureness property guarantees that the circuit never produces an incorrect output.

From the above definitions of TSC circuits it is clear that the fault set plays an important role in synthesizing these circuits. For this paper we consider the usual 'stuck-at' model. Although the stuck type fault does not model all the failure modes in a logic circuit it covers most of the common failure modes and it is simpler to work with. If all the stuck-at faults in a circuit are of the same logic value then they are called unidirectional faults. Here we are mainly concerned with unidirectional stuck-at faults and hence the set of unidirectional stuck-at faults in the checker are taken as the fault set. Moreover, we assume that the PLA structure is of AND-OR type.

The input code space, I, for the checker is the set of all k-out-of-n code words and the input non-code space, $\overline{I}$, is the set of all other n-tuples. The output code space, A, is {01,10} and the ouput non-code space, $\overline{A}$, is {00,11}.

III. K-OUT-OF-N CODE CHECKER DESIGN

Let C be the set of all k/n code words. A k/n code word is referred as a k-vertex. A binary vector $X = (x_n, x_{n-1}, \ldots, x_1)$ is said to cover another vector $Y = (y_n, y_{n-1}, \ldots, y_1)$ if $y_i \leq x_i$ for all $i = 1, 2, \ldots, n$. For example, X = (1110) covers Y = (1100). However, when $X' = (1010)$ and $Y' = (0110)$ then neither X' covers Y' nor Y' covers X'.

The fundamental idea brought about in our TSC checker design is based on the following concept. The set C_1 of all k-vertices is partitioned into m mutually disjoint subsets $S_0, S_1, \ldots, S_{m-1}$ such that the following conditions hold.

Condition 3.1: Any (k+1)-vertex, X, covers at least two k-vertices, X_1 and X_2, such that $X_1 \in S_i$ and $X_2 \in S_j$, where $i \neq j$, $0 \leq i, j \leq m-1$.

Condition 3.2: For any (k-1)-vertex, Y, covered by a k-vertex, $Y_1 \in S_i$, there must be another k-vertex Y_2, such that Y_2 covers Y and $Y_2 \in S_j$, where $j \neq i$, $0 \leq i, j \leq m-1$.

If the partition $S_0, S_1, \ldots, S_{m-1}$ satisfies the above conditions then we can design m 2-level AND-OR circuit with outputs Z_i, $i=0,1, \ldots m-1$. The output Z_i will be 1 for all the inputs in S_i, all vectors that cover S_i and 0 for all other inputs. In this realization there is a 1-1 correspondence between the k-out-of-n code words and the AND gates. Each AND gate has exactly k input lines. The outputs of the AND gates corresponding to each subset S_i are connected to the inputs of an OR gate whose ouput is Z_i.

Theorem 3.1. The circuit C satisfying the conditions 3.1 and 3.2 and outputs $Z_0, Z_1, \ldots, Z_{m-1}$ as defined above is a TSC checker circuit.

Thus, the first step in the design process is to partition the set C of k-vertices into m subsets, which satisfy the Conditions 3.1 and 3.2. The partition technique is developed in the next subsection. In the proposed design procedure we have chosen the number of subsets, m, as the smallest integer satisfying the condition

$$\min(n-k,\ k) \geq \left\lceil \frac{n}{m} \right\rceil \tag{3-1}$$

where $\left\lceil \frac{n}{m} \right\rceil$ is the smallest integer greater than or equal to $\frac{n}{m}$. The number of subsets, m, is chosen as the smallest integer in order to reduce the circuit complexity of the second PLA.

For k/2k code the value of m is 2 and thus the above circuit itself forms the TSC checker. However, for other values of m, the 1/m code is transformed into a $k'/2k'$ code using a self checking translator, similar to the one discussed in [2]. This self checking translator is nothing but a series of $2k'$ OR gates. These OR gates can be merged with the previous level OR gates. By this way we can realize a k/n to $k'/2k'$ code TSC checker using only one PLA. Another PLA, which is a $k'/2k'$ to 1/2 code TSC checker, is cascaded to the first PLA to get a k/n to 1/2 TSC checker. According to the design rule given in [2], the two level $k'/2k'$ to 1/2 code checker to be a TSC checker the value of m must satisfy the condition $2^{k'} \leq m \leq \binom{2k'}{k'}$. For m=3 or m=7, there exists no integer, k', that satisfies this constrain. Therefore the number of subsets, m, should not be 3 or 7. Whenever 3 or 7 is the smallest integer that satisfies the equation (3-1) the number of subsets, m, must be the next higher integer, say 4 or 8.

From the above discussions, we can see that the crux of the k/n code TSC checker design is the partition of the k/n code words, C, into m subsets $S_0, S_1, S_2, \ldots, S_{m-1}$, which satisfy Conditions 3.1 and 3.2. The other design rules are straightforward. Thus, hereafter, we concentrate on the partition problem.

Partition of k-out-of-n code.

We now turn our attention to the problem of partitioning the set C of k-vertices into $S_0, S_1, \ldots, S_{m-1}$ such that these subsets satisfy the two Conditions 3.1 and 3.2 mentioned before. Define a function F from C to Z_m, where $Z_m = \{0,1,2,\ldots m-1\}$, as follows.

$$\left.\begin{array}{l} F : C \rightarrow Z_m \\ \text{such that} \\ F(X) = F((x_n x_{n-1} \ldots x_1)) = \left(\sum_{i=1}^{n} i \cdot x_i \right) \bmod m \\ \text{where } \min(n-k, k) \geq \left\lceil \frac{n}{m} \right\rceil . \end{array}\right\} \qquad (3\text{-}2)$$

For example let X=(1010 0000) and m=4, then $T((1010\ 0000)) \equiv 1.8 + 1.6 \equiv 2 \bmod 4$.

The function F partitions the set of k-vertices into m mutually disjoint subset $S_0, S_1, \ldots, S_{m-1}$ such that

$$C = S_0 \cup S_1 \cup \ldots \cup S_{m-1}$$

and $S_i \cap S_j = \emptyset$ for $i \neq j$.

Moreover, if $X = (x_n, x_{n-1}, \ldots, x_1) \in S_i$

then $F((x_n, x_{n-1}, \ldots, x_1)) \equiv i \bmod m$

i.e. $F^{-1}(i) = S_i$ for $i = 0,1,2,\ldots,m-1$

Example 3.1: Consider the set of all 2-out-of-8 code words. In this case we take m=4 because $\min(n-k, k) = \min(6,2) = 2 \geq \left\lceil \frac{8}{4} \right\rceil = 2$. The code words belonging to each subset are given below.

$F^{-1}(0) = S_0 = \{$0000 0101, 0001 0100, 0010 0010, 0100 0001, 0101 0000, 1000 1000$\}$.

$F^{-1}(1) = S_1 = \{$0000 0110, 0000 1001, 0001 1000, 0010 0100, 0100 0010, 0110 0000, 1000 0001, 1001 0000$\}$.

$F^{-1}(2) = S_2 = \{$0000 1010, 0001 0001, 0010 1000, 0100 0100, 1000 0010, 1010 0000$\}$.

$F^{-1}(3) = S_3 = \{$0000 0011, 0000 1100, 0001 0010, 0010 0001, 0011 0000, 0100 1000, 1000 0100, 1100 0000$\}$.

In order to prove that the above subsets satisfy the two conditions we need the following concepts. Consider the set of integers $\{1,2,\ldots n\}$. When we take i mod m for $i=1,2,\ldots,n$, at most $\lceil \frac{n}{m} \rceil$ integers can be in the same class. For example, consider the set $\{1,2,3,4,5,6,7,8,9,10\}$. When $m = 4$, the elements corresponding to each class are as follows: 0 - class = $\{4,8\}$, 1 - class = $\{1,5,9\}$, 2 - class = $\{2,6,10\}$ and 3 - class = $\{3,7\}$. Here at most $\lceil \frac{10}{4} \rceil = 3$ elements are in the same class.

Therefore, any collection of q integers, $q > \lceil \frac{n}{m} \rceil$, taken from $\{1,2,\ldots n\}$, must contain two integers i and j such that $i \not\equiv j \bmod m$. Since this concept is useful in proving the Theorem 3.3 we can put this as a lemma.

Lemma 3.2 Any collection of q integers, $q > \lceil \frac{n}{m} \rceil$ from the set $\{1,2,\ldots n\}$ contains two integers i and j such that $i \not\equiv j \bmod m$.

Theorem 3.3 The subsets $S_0, S_1, \ldots, S_{m-1}$ obtained by the function F satisfy the Condtions 3.1 and 3.2.

Remarks:

1. For k-out-of-2k code we can take m=2 because $\min(2k-k, k) \geq \lceil \frac{2k}{2} \rceil = k$. This gives a partition equivalent to the partition obtained by Anderson and Metze [2].
2. For k-out-of-n code with $2k+2 \leq n \leq 4k$ and $2k > n \geq \frac{4}{3} k$ Marouf and Friedman use a 1-out-of-6 output TSC checker [7]. For these cases the proposed method requires only 1-out-of-4 output TSC checker.
3. For $k > \lceil \frac{n}{2} \rceil$ a two level OR-AND circuit is recommended because by this way the total number of input lines to the first level gates (OR-gates) can be reduced.
4. The set of all k-out-of-n code words forms the test set.

IV. CHECKERS FOR k/2k+1 AND k+1/2k+1 CODES

By slightly modifying the design procedure given in the previous section the TSC checkers for k/2k+1 and k+1/2k+1 codes are designed using single PLA. Even though Reddy has given a solution to this problem in [4] the partition technique used here is different from that of Reddy's and gives some insight to the design of 2-level TSC checkers for k±1/2k codes. Thus for completeness we describe the TSC checkers for these codes. Since the dual circuit of the k/2k+1 code TSC checker gives the TSC checker for k+1/2k+1 code we concentrate only on k/2k+1 code checker.

First, we partition the set, C, of k-vertices into two subsets, $F^{-1}(0)$ and $F^{-1}(1)$ using the function F for m=2 and see what the problem is. Now it is straightforward to prove that these two subsets satisfy the Condition 3.2. The proof is similar to that of Theorem 3.3. Further Condition 3.1 is also satisfied for all (k+1)-vertices except the (k+1)-vertex, (101010...01). This (k+1)-vertex has 1's at the odd index positions and zeros in all other positions. Therefore the set of all k-vertices covered by this (k+1)-vertex will be in the same subset under the defined function F. So this problem must be solved. Before giving the modified partition, we take an example and illustrate the idea behind the design procedure.

We consider the 3/7 code and partition the code words into two subsets $S_0 = F^{-1}(0)$ and $S_1 = F^{-1}(1)$ under the function F. Thus (1010101) is the only code word that does not satisfy Condition 3.1 because the 3-vertices covered by (1010101), namely (1010100), (1010001), (1000101), and (0010101), are all in S_1. To alleviate this problem, first we transfer one of these 3-vertices, say (0010101) to S_0 from S_1. Then (1010101) covers at least one 3-vertex from each of S_0 and S_1. However, now consider the 4-vertices, (0110101), (0011101) and (0010111). These 4-vertices cover elements only from S_0. To overcome this problem we transfer (0010110), (0011001) and (0100101) to S_1 from S_0. These 3 code words are obtained from the code word (0010101) by changing the odd number bit to 0 and the next even number bit to 1. Now the 4-vertices (0110101), (0011101) and (0010111) cover at least one 3-vertex from each of the subsets S_0 and S_1. Further, any 4-vertex covers at least one vertex from each of S_0 and S_1. The partition technique for an arbitrary k/2k+1 code is described below.

Let A_1 be an k/2k+1 code word of the form (0010101...10). i.e. A_1 has 1's at the odd positions except the most significant position and 0's at the even positions. Symbolically this is written as

$$A_1 = (a_{2k+1}, a_{2k}, \ldots, a_1) \text{ with}$$
$$a_j = 1, \text{ if } i = 2j-1 \text{ for some } 1 \le j \le k$$
$$= 0, \text{ otherwise.}$$

Let B_ℓ, $1 \le \ell \le k$, be a k/2k+1 code word with 1's at the $2\ell^{th}$ position and at all odd positions except at the two odd positions, $2\ell-1$ and $2k+1$. Symbolically B_ℓ can be written as

$$B_\ell = (b_{2k+1}, b_{2k}, \ldots, b_1) \text{ with}$$
$$b_i = 1, \text{ if } i = 2j-1 \text{ for some } 1 \le j \ne \ell \le k$$
$$\text{or if } i = 2\ell$$
$$= 0, \text{ otherwise.}$$

Let $A = \{A_1\}$ and $B = \{B_1, B_2, \ldots, B_k\}$. Now the mutually exclusive subsets of the code are given below

k-even

$$S_0 = (F^{-1}(0) \cup B) - A$$
$$S_1 = (F^{-1}(1) \cup A) - B$$

k-odd

$$S_0 = (F^{-1}(0) \cup A) - B$$
$$S_1 = (F^{-1}(1) \cup B) - A$$

where $F^{-1}(y)$ is the inverse image with respect to F and S-T is the difference set.

Example:

For k = 2, we have

$A = \{(00101)\}$, and

$B = \{(00110), (01001)\}$.

Thus,

$S_0 = \{(10001), (10100), (01010), (00110), (01001)\}$, and

$S_1 = \{(00011), (01100), (10010), (11000), (00101)\}$.

For $k = 3$, we have
$A = \{(0010101)\}$
$B = \{(0010110), (0011001), (0100101)\}$.

Thus,
$S_0 = \{(0000111), (0001101), (0010011), (0110001), (1000011), (1001001), (1100001), (0011100), (0110100), (1000110), (1001100), (1100100), (1010010), (1011000), (1110000), (0101010), (0010101)\}$, and

$S_1 = \{(1000101), (1010001), (1010100), (0001011), (0100011), (0101001), (0001110), (0100110), (0101100), (0011010), (0110010), (0111000), (1001010), (1100010), (1101000), (0010110), (0011001), (0100101)\}$.

Notice that $F(A_1) = 0$ if k is even, 1 if k is odd, and for all $1 \leq \ell \leq k$, $F(B_\ell) = 1$ if k is even, 0 if k is odd. It is clear that $S_0 \cap S_1 = \emptyset$ and $S_0 \cup S_1 = C$.

Theorem 4.1. The subsets S_0 and S_1 obtained above for k/2k+1 code satisfy Conditions 3.1 and 3.2.

V. CHECKERS FOR k±1/2k CODE ($k \geq 4$)

Using the similar idea as we used for k/2k+1 and k+1/2k+1 code in Section IV, we shall develop k±1/2k code TSC checker using single PLA in this section.

First of all, in Smith and Metze [6] it is pointed out that for any $n \geq R(m+1, m+1, m)$, where $R(m+1, m+1, m)$ is the Ramsey's number, and for arbitrary partition of all the m-vertices of an n-cube C into two subsets, there exists an (m+1)-vertex of C which covers no m-vertex in one of the subsets. It is known that $R(2, 2, 1) = 3$ and $R(3, 3, 2) = 6$. Thus there exists no 2-level TSC checker for 1/3 or 2/6 code. Therefore, we restrict $k \geq 4$. Since the dual circuit of the k-1/2k code TSC checker gives the TSC checker for k+1/2k code we concentrate only on k-1/2k code checker.

As we did in Section IV, we first use the function F for m=2 to partition the set, C, of (k-1)-vertices into two subsets, $S_0 = F^{-1}(0)$ and $S_0 = F^{-1}(1)$; then move around some elements from S_0 to S_1 and from S_1 to S_0 to make the partition satisfying Conditions 3.1 and 3.2. Again, before giving the modified partition, we take an example and illustrate the idea behind the design procedure.

We consider the 3/8 code and first partition the code words into two subsets $S_0 = F^{-1}(0)$ and $S_1 = F^{-1}(1)$ under the function F for m=2. Thus (10101010) and (01010101) are the only code words that do not satisfy Condition 3.1 because the 3-vertices covered by (10101010), namely (10101000), (10100010), (10001010), and (00101010), are all in S_0, and the 3-vertices covered by (01010101), namely (01010100), (01010001), (01000101), and (00010101), are all in S_1. To alleviate this problem, we transfer (00101010) from S_0 to S_1 and (00010101) from S_1 to S_0. Then we solve the coverage problem of (10101010) and (01010101). However, after this swapping we created another problem. That is, the 4-vertices (00101011), (00101110), (00111010), and (01101010), which cover the code word (00101010) just swapped, now cover only elements from S_1; and the 4-vertices (00010111), (00011101), (00110101), and (10010101), which cover the code word (00010101) just swapped, now cover only elements from S_0. To overcome this problem we transfer (00101001), (00100110), (00011010), and (01001010) from S_1 to S_0; and (00010110), (00011001), (00100101), and (10000101) from S_0 to S_1. Again after this transferring elements we created another trouble. That is, the 4-vertex (10100101) now covers only elements from S_1 and the 4-vertex (01011010) now covers only elements from S_0. To get rid of this trouble we transfer (01010010) from S_0 to S_1, and (10100001) from S_1 to S_0. After this we solved all the troubles without creating any new problem. Now we describe the partition technique for any k-1/2k code below, where $k \geq 4$.

Let $R_1 = (r_{2k}, \ldots, r_1)$ with

$$r_i = \begin{cases} 1, & \text{if } i = 2j-1 \text{ for some } 1 \leq j \leq k-1, \\ 0, & \text{otherwise.} \end{cases}$$

Let $P_1 = (p_{2k}, \ldots, p_1)$ with

$$p_i = \begin{cases} 1, & \text{if } i = 2j \text{ for some } 1 \leq j \leq k-1, \\ 0, & \text{otherwise.} \end{cases}$$

For $1 \leq \ell \leq k-1$, define $T = (t_{2k}, \ldots, t_1)$ with

$$t_i = \begin{cases} 1, & \text{if } i = 2j-1 \text{ for some } 1 \leq j \neq \ell \leq k-1, \text{ or if } i = 2\ell, \\ 0, & \text{otherwise.} \end{cases}$$

Define $T_k = (t_{2k}, \ldots, t_1)$ with

$$t_i = \begin{cases} 1, & \text{if } i = 2j-1 \text{ for some } 1 \leq j \leq k-2, \text{ or if } i = 2k, \\ 0, & \text{otherwise.} \end{cases}$$

For $1 \leq \ell \leq k-1$, define $U_\ell = (u_{2k}, \ldots, u_1)$ with

$$u_i = \begin{cases} 1, & \text{if } i = 2j \text{ for some } 1 \leq j \neq \ell \leq k-1, \text{ or if } i = 2\ell-1, \\ 0, & \text{otherwise.} \end{cases}$$

Define $U_k = (u_{2k}, \ldots, u_1)$ with

$$u_i = \begin{cases} 1, & \text{if } i = 2j \text{ for some } 1 \leq j \leq k-2, \text{ or if } i = 2k-1, \\ 0, & \text{otherwise.} \end{cases}$$

Let $V_1 = (v_{2k}, \ldots, v_1)$ with

$$v_i = \begin{cases} 1, & \text{if } i = 2j-1 \text{ for some } 1 \leq j \leq k-3 \text{ or if } i = 2k-2,\ 2k, \\ 0, & \text{otherwise.} \end{cases}$$

Let $W_1 = (w_{2k}, \ldots, w_1)$ with

$$w_i = \begin{cases} 1, & \text{if } i = 2j \text{ for some } 1 \leq j \leq k-3 \text{ or if } i = 2k-3,\ 2k-1, \\ 0, & \text{otherwise.} \end{cases}$$

Let $R = \{R_1\}$, $P = \{P_1\}$, $T = \{T_1, T_2, \ldots, T_k\}$,
$U = \{U_1, U_2, \ldots, U_k\}$, $V = \{V_1\}$, and $W = \{W_1\}$.

Example:

For k = 4, we have
R = {(00010101)} , P = {(00101010)} ,
T = {(00010110), (00011001), (00100101), (10000101)} ,
U = {(00101001), (00100110), (00011010), (01001010)} ,
V = {(10100001)} , W = {(01010010)} .

For k = 5, we have
R = {(0001010101)} , P = {(0010101010)} ,
T = {(0001010110), (0001011001), (0001100101), (0010010101), (1000010101)} ,
U = {(0010101001), (0010100110), (0010011010), (0001101010), (0100101010)} ,

$V = \{(1010000101)\}$, $W = \{(0101001010)\}$.

Notice the symmetrical structure between the odd position variables and even position variables in R_1 and P_1, T_ℓ and U_ℓ, $1 \leq \ell \leq k$, and W_1 and V_1. This symmetry will be used in the proof of Theorem 5.1. Now the mutually exclusive subsets of the code are given below.

k-even

$$S_0 = (F^{-1}(0) \cup R \cup U \cup V) - (P \cup T \cup W)$$
$$S_1 = (F^{-1}(1) \cup P \cup T \cup W) - (R \cup U \cup V) .$$

k-odd

$$S_0 = (F^{-1}(0) \cup T \cup U) - (R \cup P \cup V \cup W)$$
$$S_1 = (F^{-1}(1) \cup R \cup P \cup V \cup W) - (T \cup U) .$$

Notice that $F(R_1) = 0$ if k is odd, 1 if k is even. $F(P_1)$ always equals to 0. For all $1 \leq \ell \leq k$, $F(T_\ell) = 0$ if k is even, 1 if k is odd; and $F(U_\ell)$ always equals to 1. $F(V_1) = 0$ if k is odd, 1 if k is even. $F(W_1)$ always equals to 0. Thus, it is easy to see that $S_0 \cap S_1 \neq \emptyset$ and $S_0 \cup S_1 = C$.

Theorem 5.1: The subsets S_0 and S_1 obtained above for k-1/2k code satisfy the Conditions 3.1 and 3.2.

VI. CONCLUSION

We have shown that by partitioning the set of all k/n codes into an appropriate subsets, which satisfy certain properties, we can obtain the PLA implementation of k/n TSC checkers. The k/2k, k/2k+1, k-1/2k+1, k-1/2k, and k+1/2k code checkers require only one PLA whereas all other code checkers require 2 PLAs. Even though we have given a 2-level realization for the k/n code to 1/m code checker, using the results developed in this paper and in [14, 15], it can be shown that a multi-level checker consisting of AND and OR gates forms a TSC checker. A multi-level realization of the checker based on these results may reduce the hardware complexity and the test set, and it is worth it to pursue research in this direction.

REFERENCES

[1]. W.C. Carter and P.R. Schneider, "Design of Dynamically Checked Computers", IFIP 68, Vol. 2, Edinburg, Scotland, pp 878-883, August 1968.

[2]. D.A. Anderson and G. Metze, "Design of Totally Self-Checking Check Circuits for m-out-of-n Codes", IEEE Trans. Computer, Vol. C-22, 3, pp 263-269, March 1973.

[3]. J.F. Wakerly, "Partially Self-Checking Circuits and Their Use in Performing Logical Operations", IEEE Trans. Computer, Vol. C-23, pp 658-667, July 1974.

[4]. S.M. Reddy, "A Note on Self-Checking Checkers", IEEE Trans. Computer, Vol. C-23, pp 1100-1102, October 1974.

[5]. R. David, "Totally Self-Checking 1-out-of-3 Checker", IEEE Trans. Computer, Vol. C-27, pp 570-572, June 1978.

[6]. J.E. Smith and G. Metze, "General Rules for the Construction of m-out-of-n Totally Self-Checking Checkers", Proc. 13th Annual Allerton Conference on Circuit and System Theory, Urbana, IL, pp 704-715, October 1975.

[7]. M.A. Marouf and A.D. Friedman, "Efficient Design of Self-Checking Checkers for M-out-of-N Codes", IEEE Trans. Computer, Vol. C-27, pp 482-490, June 1978.

[8]. S.L. Wang and A. Avizienis, "The Design of Totally Self-Checking Circuits Using Programmable Logic Arrays", Digest of Papers, 9th Annual International Symposium on Fault Tolerant Computing, pp 173-180, June 1979.

[9]. N. Gaitanis and C. Halatsis, "A New Design Method for m-out-of-n TSC Checkers", IEEE Trans. Computer, Vol. C-32, pp 273-283, March 1983.

[10]. C. Halatsis, N. Gaitanis, and M. Sigala, "Fast and Efficient Totally Self-Checking Checkers for m-out-of-(2m+1) Codes", IEEE Trans. Computer, Vol. C-32, pp 507-511, May 1983.

[11]. C. Efstathiou and C. Halatsis, "Modular Realization of M-out-of-N Code Totally Self-Checking Checkers", The Thirteen International Conference on Fault-Tolerant Computing, Milano, Italy, June 1983.

[12]. S. Piestrak, "Design Methods of Totally Self Checking Checkers for M-out-of-N Codes", The Thirteen International Conference on Fault-Tolerant Computing, Milano, Italy, June 1983.

[13]. T. Nanya and Y. Tohma, "A 3-Level Realization of Totally Self-Checking Checkers for M-out-of-N Codes", The Thirteen International Conference on Fault-Tolerant Computing, Milano, Italy, June 1983.

[14]. S.B. Akers, Jr., "Universal Test Sets for Logic Functions", IEEE Trans. Computer, Vol. C-22, pp 835-839, Sept. 1973.

[15]. S.M. Reddy, "Complete Test Sets for Logic Functions", IEEE Trans. Computer, Vol. C-22, pp 1016-1020, Nov. 1973.

---

* This work is supported by the NSF grant ECS-8307450.

ANALYSIS AND IMPLEMENTATION OF STATIC AND DYNAMIC FAULT TOLERANT RING SYSTEMS*

V. OBAC RODA
Instituto de Fisica e Quimica de Sao Carlos
Departamento de Fisica e Ciencias dos Materiais
Universidade de Sao Paulo, Campo de Sao Carlos
Sao Carlos, Brasil

O. J. DAVIES
Department of Electrical and Electronic Engineering
University College London
Torrington Place, London WC1
United Kingdom

F. LOMBARDI
Department of Electrical Engineering/Computer Science
Texas Tech University
P.O. Box 4439, Lubbock, TX 79409
United States of America

ABSTRACT

In this paper the analysis and design of a particular class of fault tolerant systems is presented. These systems referred to as ring systems, are characterized by the complementary use of software and hardware for the attainment of fault tolerance. Various aspects (theory and practice for a complete implementation) are presented. These include control characterization, reliability and performability analysis, software overhead, hard core minimization and sparing.

INTRODUCTION

In applications where human life can be at risk, like nuclear power or transportation systems, or in systems that cannot be repaired, e.g., aerospace systems, reliable operation of high integrity data acquisition, is of utmost importance [1]. The extensive utilization of microprocessors for complex and critical control has enhanced the necessity of sophisticated schemes to fully exploit the powerful capabilities of these devices [2]. These capabilities are related to the extensive use of software to customize a hardware configuration for a particular application.

In this respect fault tolerance can be added in a relatively straight forward manner. However, the simplistic view of such an implementation is confronted with both theoretical and practical problems. The compact size of these devices and their complexity (approaching "large scale" attributes [4]) make the system difficult to analyze both in fault free and faulty conditions. Different and sometimes restrictive, criteria must be established to characterize attributes such as testing [5], reliability [3], performance [6], software/hardware effectiveness [7]. In practice this has been partially resolved in the formulation of new schemes and the integration of software and hardware. In the past this has been relatively successful [8], however, most of the tradeoffs are still left as a subjective choice and as a hard parameter to quantify [7].

---

*This work supported in part by the Joint Services Electronics Program at Texas Tech University under ONR grant 76-C-1136.

In this paper a new class of architecture for fault tolerant microprocessor systems is presented. The basic structure consists of a ring in which data is swapped around the transmission loop in a sequential fashion. Various aspects of fault tolerance are presented. Such structures offer a convenient alternative implementation to hardware voting in fault tolerant systems. They are a variation of the classical NMR (N modular redundancy, where N=2n+1)) structures [9], in which N modules are used to perform a certain task and the output of the system is taken from the overriding outputs. A potential application of these ring systems is front end processing for high integrity data acquisition systems, relieving the main processor of the mundane data capture of the analogue to digital conversion and of the pre-processing of the data.

A SIMPLIFIED MODEL

In order to simplify the subsequent analysis, the structures presented are considered to be subsystems, which perform a repetitive task. The subsystem model is considered connected to a main computer in a master-slave configuration. The task begins when the master issues a "Start of Conversion Pulse" (SC). The subsystem replies by reseting a "Data Valid Flag" (DAV) during the execution of the task, then it places the data on the bus and sets the DAV flag when the task is terminated. This convention is similar to the level signaling convention presented in [10].

TOPOLOGIES OF RING STRUCTURES

A ring type structure is a loop type architecture, in which the modules are linked by single unidirectional or bidirectional buses. The modules receive data from one or more sources (depending on whether these sources are replicated or not). This information is then processed and made ready for voting. The data is passed between the modules over the connective buses. Each modules then performs the voting, according to a suitable algorithm [11]. The result (agreement or discordance of the majority of the data voted) is indicated in each module by an error condition flag (EF). The logic state of the error flags is used by an output logic circuit to select the output of one of the modules as the output of the system. Part of the resources of the modules are used in the loop communication mechanism, whose functions are: transmit the results of the module processing, receive data from other modules and retransmit the data received.

The interchange of data between the modules can be visualized as a circulation of data around the loop, therefore is called "circulation." There are two possible directions of circulation: "clockwise" where (figure 1) module M1 transmits data to module M2, M2 to M3 and so on, and "anticlockwise," where circulation is in the reverse direction. In the first circulation, the module transmits the result of its processing. In the following circulations, the data previously received is transmitted by the module. This procedure continues until the modules have sufficient data for the voting [12].

VOTING STRATEGIES

Two basic voting strategies can be adopted. In the first strategy a majority voting scheme of the data from all the modules is considered. It is necessary to have N-1 circulations to fully interchange the data between the modules. For example, in a 7 module ring system as in figure 1, assuming that all the modules behave in a correct fashion and that the circulation is unidirectional (clockwise), six circulations are necessary to interchange the data between the modules.

An alternative strategy consists in reducing the number of data to be voted in each module, the limit being the comparison of data of two adjacent module. This would be possible on the following conditions:

1. The module should be able to detect an agreement condition from the data received. Situations such as an agreement between an incorrect processing result and the data received from the other modules, should not occur. These situations can be overcame by testing routines which periodically exercise the communication subsystems [1].
2. The result of the voting should be indicated correctly in each module, implying that the voting mechanism should be fault free. "Struck at" failures of the error flag indicators can be prevented by a transition level indication scheme [10].

RELIABILITY CONSIDERATIONS

For the subsequent analysis the following assumptions are necessary:

1. The modules of the system have a constant failure rate such that the reliability is exponentially distributed.
2. When the system begins its operation ($t=0$), the reliability is unity.
3. Module failures are statistically independent.
4. Only two operational states are allowed for any module, fault free or failure.

In addition to the malfunctions of the modules, faults can also happen in the interconnective links or communication subsystems. It is considered that in the case of a fault, the data received from the faulty module is incorrect and this module is unable to retransmit data from other modules. This is possible when the faulty module blocks the circulation of data through the ring. In practice faults would occur which do not affect the data transmission.

Ring systems cannot tolerate the same number of faults that can be tolerated by NMR structures with hardware voting [13]. Adopting the first voting strategy, in a N module ring system at least n+1 modules should be able to communicate in order to have in each module a majority of data to be voted. Therefore, only single faults can be masked with 100 percent confidence. For example considering the ring system in figure 2, the following error conditions can occur which enable the remaining fault free modules to have a majority of correct data:

(M1,M2), (M2,M3), (M3,M4), (M4,M5), (M5, M1)

However, error conditions such as (M1,M3) in non adjacent modules, block the communication of modules with correct operation (in this case this would be M2). In order to minimize the effect of faulty modules blocking the transmission of correct data from the other modules, the data should be circulated n/2 times in each direction. In figure 2, with M2 and M3 faulty and considering all circulation of data to be unidirectional (clockwise), after four circulations M4 and M5 will have for voting a majority of incorrect data transmitted from M3. For the circulation of data in different directions, two circulations clockwise and two anticlockwise, the fault free modules receive a majority of correct data.

The reliability of N module system (with $N=2n+1$) with all modules having equal reliability R, is

$$R_\delta = \sum_{k=n+1}^{N} R_k \qquad (1)$$

where R_k is the reliability of the system with k fault free modules ($k<=n+1$).

The first term of the above expression is the reliability of the system when all the modules are fault free:

$$R_N = R^N \tag{2}$$

As mentioned before, the condition for correct operation is that n+1 adjacent modules should be fault free. Therefore, possible faults should occur in the n remaining adjacent modules. There are N different groups with n adjacent modules. Adjacent groups have only one of the modules not in common. For example in figure 1, these are the possible 3-adjacent modules:

(M1,M2,M3), (M2,M3,M4), (M3,M4,M5), (M4,M5,M6),
(M5,M6,M7), (M6,M7,M1), (M7,M1,M2)

In each of the groups of n modules, the possible combinations of k faults which include the first module (and therefore not identical to the faults which can occur in adjacent groups) is:

$$\binom{N}{k} - \binom{N-1}{k} = \binom{N-1}{k-1} \tag{3}$$

The reliability of the ring system can be expressed as:

$$R_\delta = R^N + N \sum_{k=1}^{N} \left(\binom{N-1}{k-1} R^{N-k} \left(1-R \right)^k \right) \tag{4}$$

The reliability of ring systems employing the majority voting strategy, has been calculated for 3, 5 and 7 module structures using (4). For comparison purposes, the reliability of the equivalent NMR configurations (same number of modules and perfect voting hardware) has also been evaluated. The results are plotted in figure 3 as a function of the normalized time. It can be noticed that in a ring system, the reliability decreases as the number of modules increase. The decrease in reliability is due to the blocking condition of the faulty modules.

A similar procedure can be used to evaluate the reliability of ring systems with voting on partial data. With j the number of data to be voted, at least j adjacent modules should be fault free. The possible combination of k faults in the remaining N-j modules, which are not identical to the faults, which occurs in the adjacent modules, is:

$$\binom{N-j}{k} - \binom{N-j-1}{k} = \binom{N-j-1}{k-1} \tag{5}$$

Thus the reliability of this particular ring structure is

$$R_\delta = R^N + N \sum_{k=1}^{N-j} \left(\binom{N-j-1}{k-1} R^{N-k} \left(1-R \right)^k \right) \tag{6}$$

The evaluation of ring systems employing voting on partial data for the same cases as before, is shown in figure 4.

It can be observed that the reliability increases as smaller data is used for the voting. However, the real improvement over the TMR occurs only in the ring systems with larger number of modules. The reliability of the ring systems is very dependent on the reliability of the interconnection buses of the communication circuits. The buses, the communication circuits and the output logic circuits constitute the hard core of the ring system. For reliability considerations the hard core can be considered as being in series with the system, and affecting directly the reliability of the system. Enhancing the reliability of the hard core, the reliability of the system can be considerably improved. Buses and interconnection circuits can be replicated. A possibility is to allow the diversion of information in the case of a faulty bus by introducing extra resources for the testing of the communication channel [14]. An other alternative is to use secondary buses connecting different modules. Using multiple secondary buses connecting different modules. Using multiple secondary buses, the blocking conditions of the faulty modules can be eliminated and then the reliability of these structures becomes equal to the NMRs.

TMR RING SYSTEMS

The three module ring system (figure 5) is a particular case which has interesting characteristics. Such a structure is similar to the TMR with hardware voting and it can tolerate a single failure in one of the modules. In this configuration the modules are connected by unidirectional buses. Each module acquires data, processes it and transmits the data to the next module. The data received by the module is then compared with the results of its own processing and the result of the comparison is indicated in the form of an error flag. The error flags are connected to a logic circuit which controls the output of the system. Only one of the modules supplies the output of the system. In the system shown in figure 5, with the circulation of data clockwise, the possible outputs are:

EFa=EFb=EFc=1 output module=Ma
EFb=EFc=1, EFa=0 output module=Mb
EFa=EFc=0, EFb=1 output module=Mb
EFa=EFb=0, EFc=1 output module=Mc
EFa=EFc=1, EFb=0 output module=Mc
EFa=EFb=1, EFc=0 output module=Ma
EFb=EFc=1, EFa=0 output module=Ma
EFa=EFc=EFb=0 output module= no output.

It has been considered that a faulty module transmits incorrect data (which also includes the condition of the module being unable to send the data). For example, when there is a fault in Mb, this module sends incorrect data to Mc. Mc when comparing the data received with the result of its processing, detects a disagreement and indicates it by resetting the error flag. Mc sends the correct data to Ma, and when it compares the data received with its own data, it sets the error flag (EFa) indicating the agreement of the data. From the logic sequence of the error flags, the modules with correct operation can be identified. The module which supplies the output, is the one whose result is correct after comparison. Thus in the case of a faulty module, one of the following conditions will occur:

1. If Ma fails, Mb receives incorrect information from Ma, but Mc receives correct information from Mb. Mc supplies the output of the system.
2. If Mb fails, Ma supplies the output of the system.
3. If Mc fails, Mb supplies the output of the system.

The circuit which selects the output of the system, can be expanded to implement the dissent function with little hardware modification. The output of each module is connected to an output bus through buffers, which are selected by the output logic. The complexities of a hardware voting circuit for a TRM system and of the logic external to a TMR ring system calculated using the following expressions:

TMR with hardware voting.
NL. (3. (2 input NAND gate) + 1. (3 input NAND gate))
where NL= number of I/O lines.
If the dissent function is included, the following extra logic is required:
NL.3.(XOR gate) + 3. (NL input OR gate)
Three module ring system.
NL.3.Buffers +3.(2 input NAND gate) + 2 inverters.
By including the dissent detection, the following extra logic is required:
1 inverter + 2 input AND gate.

It can be observed that the ring system requires marginally less logic than the TMR with hardware voting (this is noticeable for higher number of I/O lines). For example the external logic required for a ring system, built with 16 bit microprocessors is 17% less than that of a TMR, and if the dissent function is implemented, it is 26% less.

IMPLEMENTATION

The TMR ring system has been implemented using as module a 8748 microcomputer with a clock frequency of 2M Hz. For this processor an instruction cycle consists of 10 states and each state is a time slot of three oscillator periods. Directly related to the performance evaluation of such system, the voting and cycle synchronization procedures have been analyzed and measured.

VOTING SYNCHRONIZATION

Since voting is directly related to the generation of a majority voting response, proper synchronized operation must be achieved to perform correctly timed data circulations.

A module can lose this synchronization due to a fault [15]. In the case of a permanent fault, a watch dog technique can be used. However a different procedure must be employed for transient faults. In this case it can be overcame either by a retry attempt (provided that the fault is detected), or a delay resynchronization on the following active cycle [11]. The ring system uses a signaling level convention [10], such that initialization and completion of the task is indicated by each module as a status output. Being synchronized, all the signals are generated by a majority voting on the status of the modules. Due to variations of the transducers or to quantization errors during the conversion, the modules can receive legitimate, but different inputs.

Limits of validity can be established for the comparison. This is done either using a round off method before the processing, or establishing a maximum acceptable difference between the data [16]. If rounding off or a similar technique is used, the data to be processed is identical for all the modules (assuming that the data acquisition is performed correctly and that the transducer is fault free). The processing time should be the same in each module and locked step clocks can be used [15]. However, a different procedure must be established, if the maximum acceptable difference between the comparison data is employed. The processing time between the modules can differ depending on the input values. In order to keep the modules in synchronization, it is possible to set a cycle counter at the beginning of the data acquisition process; the counter triggers the comparison after a certain number of cycles (maximum execution time for the data acquisition). The main inconvenience of this method is that the modules remain idle for some time before the comparison takes place. Another synchronization possibility that has been explored is a handshake scheme. The module transmitter sets a flag when the data is read for comparison. The receiver module acknowledges the transmission. During the handshake the following conditions occur:

1. Module transmitter: send data #1; receive acknowledge #2.
2. Module receiver: receive data #3; acknowledged data received #4.

After #2 and #3 an internal delay is set in the modules to allow the handshake to be completed and for the slowest module to respond. The delay should be equal to the maximum difference between the slowest and fastest data capture and processing. If during the allowed delay, the handshake is not completed (due to a module failure), the module proceeds its operations performing the comparison, resetting the status flag and waiting for the start command. If in the next active cycle the faulty condition persists, the module either turns off the delay mechanism for the rest of its operation (after this, the comparison is performed immediately after one of the handshake conditions #2 or #3 takes place), or a request for a test is validated. For the experimental TMR ring system, the following execution times have been recorded:

1. Transmission of data to another module: 40 microsecond.
2. Handshake: 555 microsecond.
3. Comparison of data: 60 microsecond.

CYCLE SYNCHRONIZATION

Previously a master-slave configuration has been analyzed as the basic model of the ring system. The close operation of the master and a slave can be implemented by an interrupt service routine to provide synchronization for the start signal. In [15] a hardware synchronization unit is suggested. This unit performs a majority voting on the status output of the modules followed by a delay, which allows the slowest module to complete the task. A similar scheme can be implemented in software. In a software implementation each module monitors the status output of the other modules and a software delay loop is activated when the majority of the modules completes the conversion. For the TMR ring system the software cycle synchronization has been recorded to take 195 microsecond.

HYBRID RING SYSTEM

A ring system utilizing standby spares, is referred to as hybrid ring system. The provision of spares to the system must obey the following conditions:

1. The basic ring structure must be maintained (modularity); this implies that all initial modules and spares must be identical in their operations.
2. When replacing a faulty module by a spare, all the input/output connections and interconnection links must be replaced.
3. The overhead incurred in the switching of the spares must be within the time constraint of the system.

The possibility of any spare taking any position in the loop (total assignment) is impractical for ring systems because it would require a rather complex switching mechanism. In [13] a bypassing spare mechanism is proposed together with a decentralized control. While avoiding the hard core characteristic, it requires the modules to be able to indicate their condition (diagnosability) in the presence of faults. This implies that each module should have a self testing mechanism which must be either fault tolerant or self checking.

An alternative to this design can be described as a replacing system. The switching of the spares for each module is controlled by a counter & decoder circuit (figure 6). When a spare is switched on into the ring, the data is routed through the spare by the use of buffer or multiplexers. At the beginning of the system operation, the counters are reset and the system starts operation with the initial modules switched into the ring (Swa1-SWb1-SWc1=1). During the normal operation of the system, when all the modules are fault free (EFa'=EFb'=EFc'=1), the control signal for the ring system (BC and EC') are identical to the SC and DAV' signals of the master computer. When one of the module malfunctions (this is indicated by one of the error flags being zero), the controller generates a BC command to the ring system and at the same time the error condition is memorized by FF1. The BC command is generated in the controller by a monostable triggered by the high to low transition of EC'. If the fault is a transient, the next EC' transition resets FF1 and the system proceeds in its normal operation. If the fault is permanent, the fault indication is repeated in the following conversion cycle and this repetition sets FF2. This enables the counter correspondent to the faulty module to be incremented and a spare to be switched on line into the ring to replace the faulty module.

CONCLUSIONS

The basic principles of operation of static and hybrid dynamic ring systems have been presented. The basic TMR scheme has been constructed and procedures like voting synchronization, cycle synchronization and voting have been experimentally evaluated. A new design for the insertion of spares into the system to satisfy a modularity growth has been presented.

REFERENCES

[1] Kraft, G.D. and Toy, W. N., "Microprogrammed Control and Reliable Design of Small Computers," Prentice Hall, Englewood Cliffs, 1981.

[2] Siewiorek, D. P. and Swarz, R. S., "The Theory and Practice of Reliable System Design," Digital Press, Bedford, 1982.

[3] Davies, O. J. and Lombardi, F., "Toward More Reliable Microcomputer Systems," Proc. IEE Colloquium on Reliable Systems for Microprocessor Control, Digest 69, pp. 1-3, London, 1980.

[4] Saeks, R. and Palmer, J. D., "The World of Large Scale Systems," IEEE Press, 1983.

[5] Annatore, M. A. and Sami, M. G., "An Approach to Functional Testing of Microprocessors," FTCS, pp. 158-164, 1982.

[6] Ferrari, D., "Computer Systems Performance Evaluation," Prentice Hall, Englewood Cliffs, 1978.

[7] Wulf, W. A., "Reliable Hardware/Software Architecture," IEEE Trans. on Software Eng., Vol. SE-1, No. 2, pp. 233-240, 1975.

[8] Bernhardt, D. and Schmitter, E., "Design and Implementation of a Fault Tolerant Microcomputer System," Microprocessors and Microsystems, Vol. 5, No. 4, pp.153-157, 1981.

[9] Avizienis, A. "Fault Tolerance and Fault Intolerance, Complementary Approaches to Reliable Computing," 1975 Conf. on Reliable Software, pp. 458-464, 1975.

[10] McConnel, S. R. and Siewiorek, D. P., "Synchronization and Voting," IEEE Trans. on Comp., Vol. C30, No. 2, pp. 161-164, 1981.

[11] Obac Roda, V., Davies, O. J., Harvey, H. E. and Lombardi, F., "Analysis and Implementation of System Procedures for Software Implemented Fault Tolerant Microcomputer Schemes," IEEE 2nd Annual Phoenix Conference, pp. 123-127, 1983.

[12] Pease, M., Shostak, R. and Lomport, L., "Reaching Agreement in the Presence of Faults," Journal of the ACM, Vol. 27, No. 2, pp. 228-234, 1980.

[13] Lombardi, F., "Modelling and Analysis of Fault Tolerant Microcomputer Systems," Ph.D. Thesis, University of London, 1982.

[14] Obac Roda, V. and Lombardi, F., "Software Implementation of a Disagreement Detector for Duplex Hybrid System," Eurocon 82 Electrotechnics Conference, pp. 673-678, North Holland Publ. Co., 1982.

[15] Davies, D. and Wakerly, J. F., "Synchronization and Matching in Redundant Systems," IEEE Trans. on Comp., Vol. C27, No. 6, pp. 531-539, 1978.

[16] Hecht, H.,"Fault Tolerant Software for Real Time Application," ACM Computing Surveys, Vol. 8, No. 4, pp. 391-407, 1976.

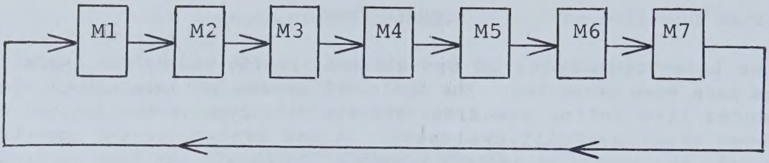

M1
M2
M3
M4
M5
M6
M7

Figure 1: 7 module ring system.

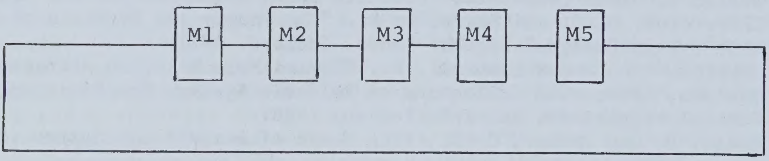

M1
M2
M3
M4
M5

Figure 2: 5 module ring system (bidirectional).

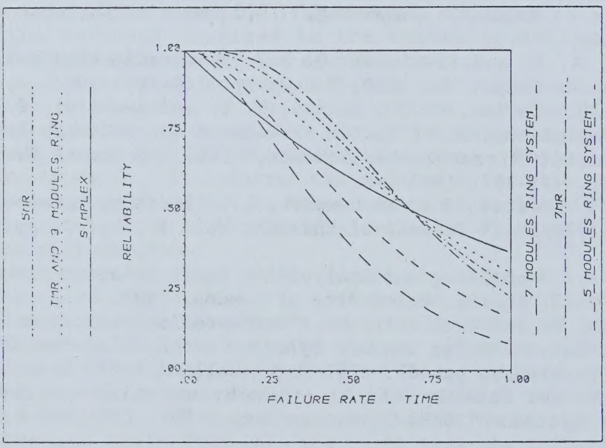


Figure 3: Ring system reliability vs normalized time.

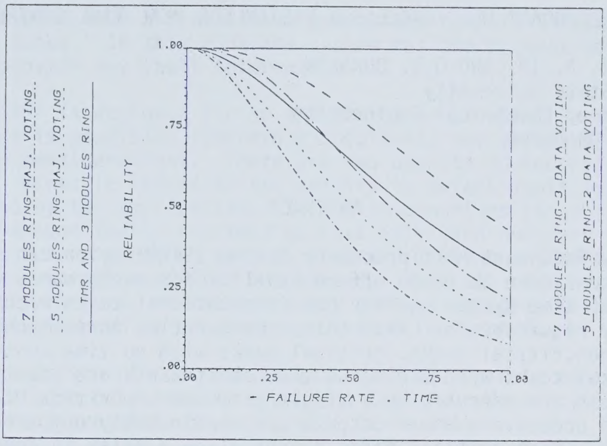


Figure 4: Ring System reliability (partial data voting).

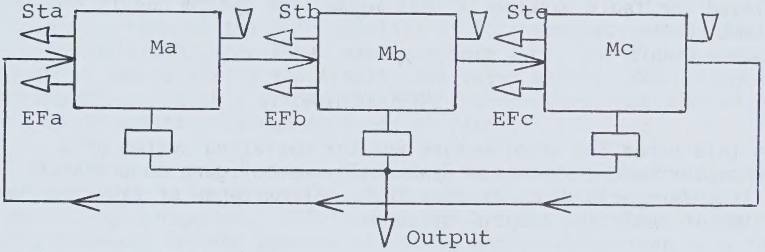

Sta
Stb
Stc
Ma
Mb
Mc
EFa
EFb
EFc
Output

Figure 5: TMR ring system.

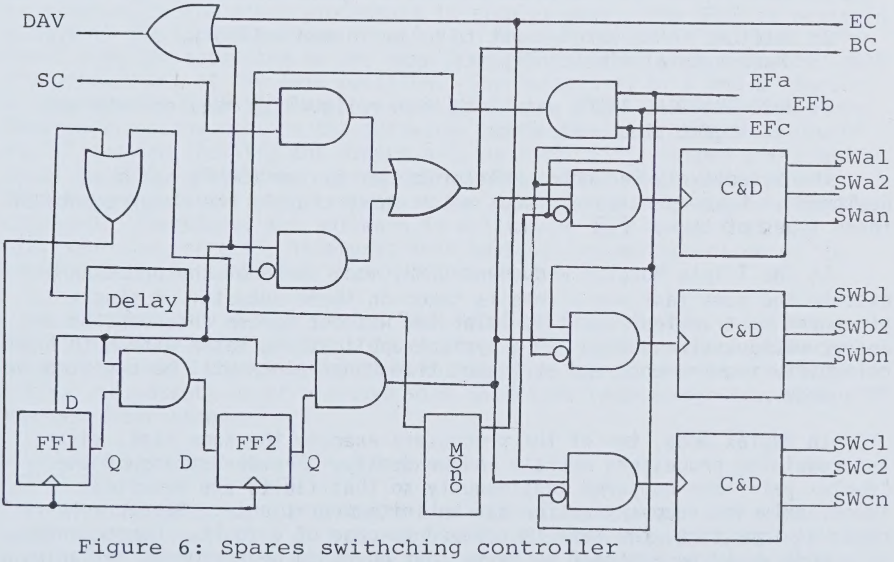

DAV
SC
EC
BC
EFa
EFb
EFc
SWa1
SWa2
SWan
C&D
Delay
SWb1
SWb2
SWbn
C&D
D
FF1
Q
D
FF2
Q
Mono
SWc1
SWc2
SWcn
C&D

Figure 6: Spares swithching controller

A SELECTIVELY REDUNDANT MULTIPROCESSOR SYSTEM FOR REAL TIME CONTROL

F. OZGUNER, A. IP, AND O.V. DUNGERN
The Ohio State University
Department of Electrical Engineering
Columbus, OH 43210

ABSTRACT

A Selectively Redundant Multiprocessor System (SRMP) which can dynamically reconfigure to trade off reliability for performance is described. A real time system usually has computational tasks with different reliability requirements. With this consideration three types of tasks, namely noncritical tasks, critical tasks with no time constraints and most critical tasks with stringent time constraints are identified. Noncritical tasks are executed by a single processor. Critical tasks are executed by two processors whose outputs are continuously compared for fault detection. Most critical tasks cannot allow a delay in computations due to recovery. Therefore a fully triplicated configuration with voting is employed for fault masking in this mode. The system consists of four subsystems.

INTRODUCTION

In this paper the architecture and the operating system of a multiprocessor system which can dynamically reconfigure to trade off reliability for performance is described. Three types of tasks can be identified in real time control systems.

1) NONCRITICAL TASKS with no particular fault tolerance requirements,

2) CRITICAL TASKS which ought to be performed reliably, but do not have special time contraints, and

3) MOST CRITICAL TASKS with both high reliability requirements and stringent time contraints.

The Selectively Redundant Multiprocessor System (SRMP) has been designed to have operational modes which match closely the requirements of these types of tasks. [1]

In the Triple Modular Redundant (TMR) mode three of the processors execute the same task and a vote is taken on their outputs. A single permanent or transient fault is tolerated without system intervention and hence, without time delay. In a typical application, tasks with both high reliabiliy requirements and stringent time constraints will be executed in this mode.

In duplex mode, two of the processors execute the same task, while the remaining processors operate independently. The operations of the "duplex pair" are compared continuously so that faults are detected immediately and recovery action can be initiated at once. Since recovery requires some time, the task is delayed in case of a fault. Consequently, this mode would be assigned to tasks that ought to be performed reliably, but do not have any stringent time constraints.

In simplex mode all available processors operate in parallel on different tasks. In this mode the system has the highest performance, but does not provide any fault tolerance.

The SRMP is designed for an industrial environment where on-line maintenance is possible. Therefore modularity and software transparency were major considerations. There are two aspects to reconfiguration in the SRMP. First it refers to the ability to switch fault tolerance modes as required by the application. Second it comprises the ability to configure around faulty modules, so that only good ones are employed. Continuous reconfiguration ensures that a different set of processors serve in a critical mode (TMR or duplex). Thus all of them are continuously checked, so that a faulty one can be identified and subsequently ignored by the healthy core of the system.

SRMP consists of four subsystems which can operate as independent computers or can cooperate to execute a task in redundancy. Only one redundant configuration can exist at a given time.

ARCHITECTURE

Each subsystem of the SRMP consists of four modules, namely the processor module (PM), the local memory module (LM), the global memory and input/output module (GMIO) and finally the voter module (VM). They are interchangeable, not only with respect to the hardware, but also with respect to permanently stored programs or data.

Figure 1 shows how a single subsystem is connected to the system bus. SB1 refers to the primary bus, which is controlled by the processor of that particular subsystem. SB2 through SB4 are the secondary buses, which are only observed for the purpose of voting and/or comparison in a fault tolerant mode. They are also connected to the GMIO module so that it can be accessed by the other processors in simplex mode. The SRMP is designed such that all subsystems are identical. The system bus is twisted as it leads from one subsystem to the next (Fig. 2), so that each subsystem sees its primary bus at the same position. The letters A, B, C and D identify a subsystem in a particular physical location. This arrangement has some important consequences on the software. Each subsystem refers to itself as "1" and can identify the others only as lower ("2"), upper ("4") and opposite ("3") neighbors. Hence, all configuration codes and error signals are relative with respect to the subsystem and cannot be directly compared. Therefore, the software is written in such a way that the processor can correctly interpret this sort of information based on the knowledge of the subsystem that has generated it.

A preliminary design of the SRMP has been completed by using the MOTOROLA 6809 microprocessor. An eight-bit processor has been preferred, because the main objective of this research was to explore the characteristics and potentials of a system with selective redundancy independent of the processor size.

Processor and Local Memory Modules

The processor module contains the 6809 microprocessor and some support logic including an interrupt controller and a synchronizing clock unit. The local memory module is only accessed by the processor in the same subsystem and therefore is connected to the primary system bus. It contains all application and system programs and provides some read/write memory for scratchpad activities.

Voter Module

The voter module performs the voting and fault masking in TMR mode and the error detection in both redundant modes. The voter module is connected to all four system buses (Fig. 1). Logic values appearing on these buses can just be observed for the purpose of comparison or voting. In TMR mode the outputs of the voting circuits are placed only on the primary bus, since each of the subsystems is equipped with its own voter. Fig. 3 is a block diagram illustrating the pertinent signals going into and out of the voter module. The error register is set by the error signal decoder to record the disagreeing module in TMR execution. Its contents are used later for fault confinement and reconfiguration purposes. The function of the voter module is determined by the contents of the configuration register. One bit is associated with each subsystem, which is set to "1" to route the bus of that particular subsystem to the voter and fault detector. A "0" deselects this bus. Three of the four bits ought to be set to "1" in TMR mode, while at least two must be set for disagreement detection. Whenever, the system is reconfigured, the associated processor writes a new bit pattern into this location.

The logic diagram of the voter and disagreement detector is shown in Fig. 4. Product terms $1 \cdot 2$, $\overline{1} \cdot \overline{2}$,..., $3 \cdot 4$, $\overline{3} \cdot \overline{4}$ are generated. Both the disagreement signals and the voter output are obtained by OR'ing a different subset of the product terms. Thus this unit can easily be implemented by using PLA's. The six disagreement signals (D12-D34) indicate disagreements between duplex pairs and are also used to identify the disagreeing processor in TMR mode.

The error signal decoder (Fig. 3) recognizes catastrophic errors (those requiring immediate system reconfiguration) in both the duplex and the TMR mode. In duplex configuration a single error is catastrophic, while a double error is catastrophic in TMR mode.

Erroneous subsystems are identified in the following way in TMR mode. In case of a single error two disagreement signals go on, as the erroneous subsystem disagrees with both of the other subsystems. For example, subsystem 2 is faulty, if two out of the signals D12, D23 and D24 are asserted. If three of the four disagreement signals are asserted in the TMR mode or a disagreement is detected in the duplex mode a Redundant Mode Failure (RMF) interrupt is activated.

Four levels of synchronization have to be considered, namely at the clock, machine cycle, instruction and real time frame levels. Synchronization at the three lower ones becomes necessary, because a hardware voting scheme has been selected for the SRMP, which requires that the participating processors be in close lockstep. The voter is responsible for getting the adherent processor module into lock step synchronism with the cooperating processors. This is done by comparing the E and Q signals [2] with those of the cooperating processors. A difference in phase would assert the stretch (STCH) signal, which stretches the E and Q signals of the adherent processor module, until all cooperating processors have the same relative phase.

As seen from Fig. 1 off line voting is employed. The major advantage of such a scheme is that in case of a failure the voter can be removed while the subsystem can still operate in simplex. Furthermore, it poses no delay in simplex operation.

Global Memory and I/O Module

Interprocessor communication is through global memory which is segmented and protected from erroneous accesses. Each subsystem contains one of the four segments of global memory. It is very suitable to combine the global memory (GM) and I/O interface into one module (GMIO module). I/O addresses are located in the address map of the global memory and consequently access to an I/O location is similar to an access to GM. This simplifies both the software and the hardware implementation.

Each GMIO can be accessed in three different modes: absolute, relative, and redundant. The absolute mode enables a processor to access a fixed location in any of the GMIO modules. This mode is implemented for accessing memory mapped I/O locations. The relative mode allows a processor to access a location relative to its position. This mode allows interprocessor communication to be performed without assigning fixed identities to the processors. This implementation is essential as all processors use the same supervisor program. The difference between the absolute and relative accessing mode is the address assigned to a given location. With the absolute mode, a given location has the same address for all processors, while the same location has different addresses in relative mode depending on which processor is doing the access.

During redundant operation all segments are simultaneously accessed in parallel. In order to allow addresses to be compared and/or voted upon, all segments have to assume the same address range. This access mode addresses simultaneously the same relative location in each GMIO module so that data manipulated in a TMR or duplex task are updated in all modules at the same time.

In the absolute accessing mode, the four GMIO modules can be considered to be contiguous to form a 4 K-byte block beginning with GMIOA, ending with GMIOD. The suffixes A, B, C, and D in Fig. 5 are attached for the sake of explanation. Each processor holds a relative view and identifies itself as PM1. Specifically, the identifications of the processors from PMA's point of view are

PM1 = PMA, PM2 = PMB, PM3 = PMC, and PM4 = PMD.

For PMB, the identifications are

PM1 = PMB, PM2 = PMC, PM3 = PMD, and PM4 = PMA.

Because of the different accessing modes, mode dependent address decoders are provided. Starting addresses of each GMIO module for different accessing modes are shown in Fig. 5.

Since more than one processor can potentially access a GMIO module, the latter must have an arbiter which resolves access contention. A "batch" arbitration scheme has been devised, which arrives at an acceptable, yet simple, solution. Processors in a redundant mode are given absolute priority as they are in tight synchronization.

THE SUPERVISOR PROGRAM

As a first step in the development of an operating system, a supervisor program has been developed and debugged with the HP 64000 development system. Since a separate identical supervisor is needed for each subsystem routines were debugged with a 6809 emulator by simulating the environment. The configuration registers and error registers are memory locations and the development system permits the user to change the contents of memory locations and single step. Thus dynamic behavior of the other subsystems was simulated by running the supervisor up to a certain point and then altering memory locations.

The system software is constructed to maintain the symmetry and autonomy of each subsystem. Each subsystem has a copy of the system software. This makes each subsystem fully capable of operating alone until cooperation from other processors is needed for execution of a reliable task. All processors have all application programs as well.

The supervisor program is designed to provide the following services:

1) scheduling
2) interprocessor communication
3) configuration and synchronization of core processors for redundant operation
4) rollback point setting
5) duplex fault handling
6) fault confinement

The supervisor program and all application programs are to be stored in ROM for fast access and promptness to respond to environmental changes. Since all memory assignment can be made before the system is put into operation, memory management is left out in the supervisor program.

I/O management is also not included with the assumption that the system interacts with its environment through sensors and actuators, and memory mapped locations are reserved in the GMIO modules.

Scheduling

The SRMP considers three types of tasks: (i) periodically executed tasks, (ii) event triggered tasks, and (iii) low priority tasks which do not have any time constraints. Types (i) and (ii) can be executed in any of the three modes whereas type (iii) tasks are always executed in simplex.

Owing to the predictable nature of the tasks in type (i), a schedule can be planned before the system is implemented. Time is divided into frames and a set of time frames can then be determined to satisfy the execution rates of all tasks [3]. Since these tasks are to be executed periodically when the system is running, the set of time frames is repeated after the whole set is executed. One time segment is defined as the time required to execute the set of time frames. The schedule determined for each time frame forms the local task list of each processor module. Fig. 6 illustrates an example of time frames in one time segment with the corresponding local task list in each frame. One important point that must not be overlooked is that there must be some idle time in every time frame for the processor to handle event triggered and low priority tasks.

Tasks of type (iii) are not scheduled into the time frames since they have no time constraints. All tasks that fall into this category are collected in a single list, the global task list. These tasks can be executed by any processor in the system as long as the processor finishes its local task list and the time for its current frame is not exhausted. Schedule for tasks of type (ii) is not predictable. A processor decides if a certain task should be executed after checking for the occurrence of the appropriate events.

Each processor has its own scheduling loop and its local and global task lists. The local task list is used only by the individual processor, while one copy of the global task list is chosen to be shared by all processors.

Interprocessor Communication

Each processor designates an area in its adherent GMIO module as mailboxes for other processors. It sends messages by writing into the mailboxes. Fig. 7 shows the mailboxes. A processor can only read messages in other processors' memories and is not allowed to write in them. The sender identifies its receiver by dropping the message into the area reserved for that processor. Similarly, the receiver identifies the sender by knowing the location from which the message is read. Fig. 7 depicts the mailboxes in their absolute addresses, starting with subsystem A. Each processor reads messages after each task.

Global Task List

Unlike the local task list, all processors share a chosen global task list. The purpose of sharing a single global task list is to provide fair service to unscheduled simplex tasks. By sharing a single list, each processor picks the next task down the line, each task waits a quarter of the time that it would if the list is scanned by one processor. If a processor is detected as being defective, all tasks will still be executed, just at a slower rate than if all processors are functional. One implicit assumption on the global tasks that supports the sharing of one list is that data concerning a global task is not passed to another subsytem. The reason is to prevent data contamination.

Fig. 8 depicts the structure of the global task list and the procedure of accessing the list. Before accessing the list, the processor is liable to set a lock. If locking is successful the processor can manipulate the contents of the list. After knowing the task to execute, the holder of the list is responsible to unlock the list. But if the holder of the list has a transient or hard failure and is unable to reset the lock, other processors would be kept waiting forever. To prevent this occurrence, the lock is designed to be resettable by waiting processors.

In its unlock state, the lock of the list has value FFH. A processor tries to lock the list by incrementing the contents of the lock. The first processor that attempts to lock the list would change the lock to 00H with carry set. A set carry flag indicates the locking is successful. Since incrementing the contents of a memory location is an indivisible instruction, only one processor would be successful in setting the lock. Further attempts would simply increase the contents of the lock with the carry flag reset. It is obvious that the contents of the lock will eventually reach FFH. The next processor that attempts to lock will be successful. Therefore, the list will be granted to another processor after 256 locking attempts, no matter if the first user has unlocked the list.

Configuration and Synchronization of Core Processors

Synchronization of subsystems is called for only when a task that requires redundant operation is pending, and synchronization concerns only the subsystems involved in the redundant execution. The redundant core is configured as follows in the case of event triggered tasks. The user program informs the supervisor of the id of the response task and the mode of execution in a specific location -RED. The six bit task id field identifies a slot in a list, the redundant task list, which contains the starting address of tasks that may be executed in redundancy. The id is the offset from the beginning of the list. Three different modes can be specified: 01 -simplex, 10 -duplex and 11 -TMR. If duplex or TMR mode is specified, the processor needs to find enough partners to satisfy the requested mode. A special list, the timestamp list, is established in each GMIO module for broadcasting the decisions of the corresponding PM. The timestamp of a task is the reading of the subsystem clock when the processor detects the triggering event of the task. A valid timestamp is one with a non-null value. A zero value indicates that the processor has not detected the triggering event of the task. Timestamps are used for determining the core processors and which task to execute if more than one is pending. Each processor reads timestamps of other processors. However, when a processor attempts to read a timestamp from a GMIO module, it has to be sure that the corresponding subsystem is functional. When three or more processors consent to execute the same task, the core processors can be chosen according to their timestamps. This in effect provides dynamic and random core configuration so that each processor has equal probability to be involved in a redundant execution. Processors with the earliest timestamps are picked to form the core. Agreement from three processors is also necessary for duplex tasks as one processor is needed as a backup in case the two core processors give disagreeing outputs.

The PREP message is sent when a processor has decided upon a task. It is a call for synchronization to other core processors. The voter configuration is part of the message. The PREP message serves both as a call to synchronization for a task as well as an acknowledge to a call. Subsequent to sending the PREP message, a processor will not return to scheduling for another task until the requested reliable task is done. It keeps on checking messages until a PREP message is received from all functional processors. Then it can vote on the voter configuration according to the received decisions. By knowing the voter configuration and the mode of the task requested, the processor can find out if it is included in the core. If it is, it simply sets up the voter and executes a SYNC instruction [2], waiting for synchronization with the other core processor(s).

If all processors schedule the same task and the core processors go into the SYNC state successfully, the voter modules provide an interrupt. Upon receiving the interrupt, the core processors exit the SYNC state and start to operate in lock step synchronism. This continues until the end of the task. At the end of the reliable task, each processor relinquishes synchronization by disabling the voter module outputs. A FINISH message is broadcasted for the task. This message is used to inform non-core processors that they can schedule tasks again.

Duplex Fault Handling

It is well known that duplex execution provides, fault detection, but not fault location. To recover from a detected fault, rollback is a common practice [4]. The rollback point set up feature is implemented for this purpose. In SRMP rollback is performed in TMR instead of duplex as TMR operation is a standard feature of the system.

REFERENCES

[1] O.V. Dungern: "A Multiprocessor Architecture with Selective Redundancy for Real Time Control," M.S. Thesis, The Ohio State University, Department of Electrical Engineering, Columbus, OH, June 1983.

[2] Motorola Semiconductors, MC6809 Data Sheet, Austin, TX.

[3] S.T. Allworth: Introduction to Real Time Software Design, Springer Verlag, New York, NY 1981.

[4] D.P. Siewiorek, R. Swarz: The Theory and Practice of Reliable System Design, Digital Equipment Corp., 1982.

[5] A. Ip: "A Supervisor Program for a Selectively Redundant Multiprocessor System," M.S. Thesis, The Ohio State University, Dept. of Elec. Eng., Columbus, OH, August, 1983.

[6] L. Maher, S.J. Larimer: "Continuous Reconfiguration in a Multi-Microprocessor Flight Control System," Proc. of the AGARD Avionics Panel on Tactical Airborne Distributed Comp. and Networks, pp. 9.1-9.9, June, 1981.

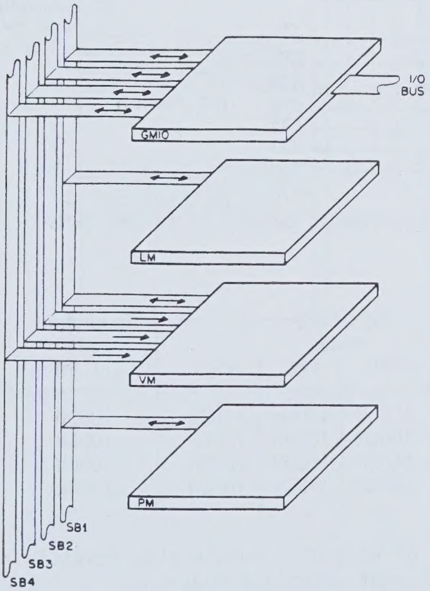


Figure 1. A subsystem of the SRMP

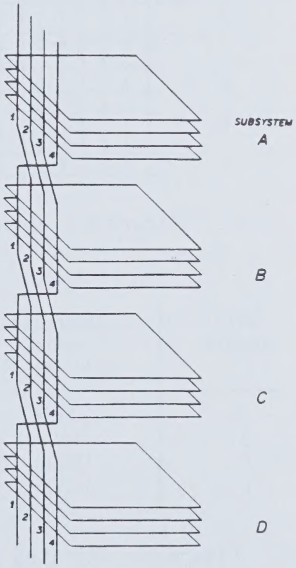


Figure 2. The twisted system bus of the SRMP

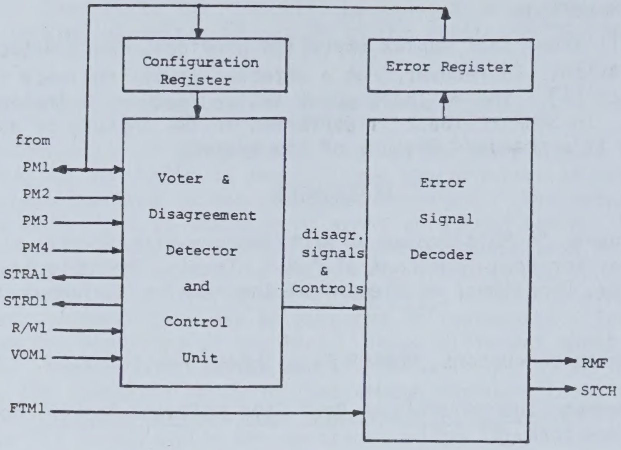

Configuration Register
Error Register
from
PM1
PM2
PM3
PM4
STRA1
STRD1
R/W1
VOM1
FTM1
Voter & Disagreement Detector and Control Unit
disagr. signals & controls
Error Signal Decoder
RMF
STCH

Figure 3. Block diagram of the voter module

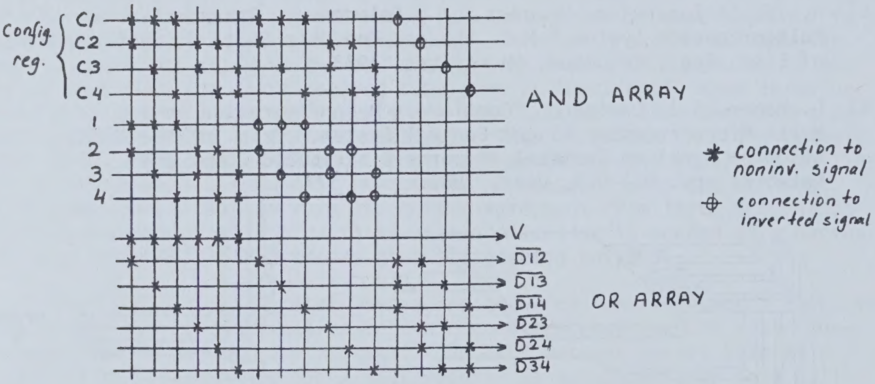

Config. reg.
C1
C2
C3
C4
1
2
3
4
AND ARRAY
✱ connection to noninv. signal
⊕ connection to inverted signal
V
$\overline{D12}$
$\overline{D13}$
$\overline{D14}$
$\overline{D23}$
$\overline{D24}$
$\overline{D34}$
OR ARRAY

Figure 4. Voting and disagreement detection in the SRMP

GMIO module	Absolute mode (all PM's)	Relative mode				Redundant mode (all PM's)
		PMA	PMB	PMC	PMD	
A	1000H	1000H	1C00H	1800H	1400H	1000H
B	1400H	1400H	1000H	1C00H	1800H	1000H
C	1800H	1800H	1400H	1000H	1C00H	1000H
D	1C00H	1C00H	1800H	1400H	1000H	1000H

Figure 5. Starting addresses of each GMIO module with respect to each PM for different accessing modes

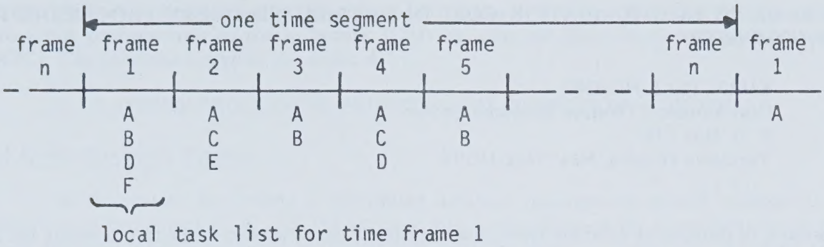

one time segment
frame n
frame 1
frame 2
frame 3
frame 4
frame 5
frame n
frame 1
A B D F
A C E
A B
A C D
A B
A
local task list for time frame 1

Figure 6. An example of time segmenting with local task lists for individual time frames

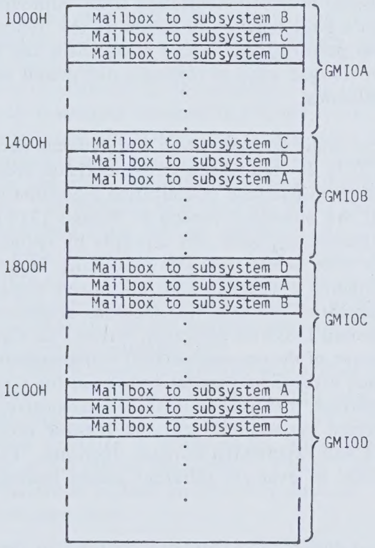

1000H
Mailbox to subsystem B
Mailbox to subsystem C
Mailbox to subsystem D
GMIOA
1400H
Mailbox to subsystem C
Mailbox to subsystem D
Mailbox to subsystem A
GMIOB
1800H
Mailbox to subsystem D
Mailbox to subsystem A
Mailbox to subsystem B
GMIOC
1C00H
Mailbox to subsystem A
Mailbox to subsystem B
Mailbox to subsystem C
GMIOD

Figure 7. Layout of mailboxes in absolute mode

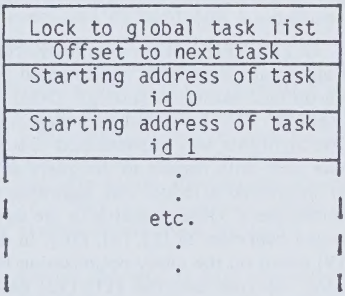

Lock to global task list
Offset to next task
Starting address of task id 0
Starting address of task id 1
etc.

Figure 8. Format of Global Task List

MINIMIZING COMMUNICATION COST IN DISTRIBUTED QUERY PROCESSING IS NP-COMPLETE

KUAN-TSAE HUANG
IBM Thomas J. Watson Research Center
P. O. Box 218
Yorktown Heights, New York 10598

Abstract: In distributed database management systems, the efficiency of query processing has a strong influence on the performance of the systems. It is recognized that deriving an optimal distributed processing strategy, in the sense that some cost is minimized, is a very difficult problem. This paper show that under three different objectives finding an optimal query processing strategy in distributed databases is an intractable problem.

1. INTRODUCTION

In distributed database management systems, the efficiency of query processing has a strong influence on the performance of the systems. Query processing in a distributed system is different from that in a centralized system. In distributed system, data are stored at computers which are geographically separated; hence query processing involves some local data processing and the necessary data transmission over communication links. Although both type of operations will introduce time delays, for a large network of databases the transmission delay plays the major role in the overall system performance. Many researchers and system developers have considered these facts and have derived ways of finding a distributed processing strategy for data processing and data transmissions.

In POLYPHEME prototype system, both static decomposition and dynamic decomposition algorithms have been implemented [16], [17]. It used an adaptative decision process to find an optimal extension strategy. Other research in dynamic optimization algorithm was by Baldissera et. [1]. In SDD-1 prototype system, An enhanced version of Wong's [21] greedy heuristic algorithm was implemented [3]. It exploited the semi-join concepts by reducing the database state with respect to the query at each node before final join processing. Huang [13] further developed a model and algorithm for finding a semijoin and join mixed strategy for distributed query. Other research in the development of full-reduction methods of processing a distributed query are in [2], [4], [10]. In distributed INGRES prototype system, An algorithm [7] [19] based on the query optimization technique of decomposition [20] is implemented and evaluated. Hevner and Yao [11] [12] developed algorithms Parallel and Serial for a class of simple queries. They extented to a heuristic algorithm that uses an improved exhaustive search for the general quries. Chu and Hurly [5] defined the solution space of feasible processing strategies for distributed queries and developed and exponential optimal algorithm. Pelagatti and Schreiber [18] developed a multilevel model to evaluate different access strategies by using integer programming technique.

It is recognized that deriving an optimal distributed processing strategy, in the sense that some cost is minimized, is a very difficult problem and all algorithms that have been proposed to date are heuristic. This paper show that under three different objectives finding an optimal query processing strategy in distributed databases is an intractable problem.

The organization of this paper is following. In the next section, we describe the query processing problem, review complexity theory and list the three NP-complete problems we need to use for proving our results. In section 3, we prove the NP-complete results for the query processing problem with a bounded number of nodes in each subtree (QPBS) by using the satisfiability problem (SAT), for the query processing with a bounded number of subtrees

(QPBT) by using the bounded component spanning forest problem (BC), and for the query processing problem with maximum benefit (QM) by using the exact cover by 3-sets problem (X3C). The conclusion is given in section 4.

2. QUERY PROCESSING PROBLEM AND COMPLEXITY THEORY

2.1 Query Processing Problem

In this paper, we assume a distributed database management system consists of a collection of computers $S_1,S_2,...,S_n$ at different sites interconnected by a computer network. Each computer, known as a node in the network, provides storage capability for database and processing capability for execute query functions over the database. The database is distributed across the network without redundant copy. The data at each node is accessiable to the users from any node of the network. Data is logically viewed in the relational model. Without lost of generality, we assume each node only consists of one relation.

Data transmission in the network is via communication links. We adapt the assumption that data transmission costs dominate tha local processing costs. We assume that the transmission cost to send one byte of data from node i to node j is known and equal to to c_{ij}. Thus, the transmission cost is a linear function of data volume to be transmitted. For each query, we process all possible subqueries invovling data at a single node first. The effect of this local processing is to reduce the amount of data that needs for futher processing. We further assume that all possible semijoin operations is performed to reduce the database state into minimal state as in [2], [4], [3]. Hence, after these processes, the size of each resulting relation can be computed as the product of the number of tuples and the sum of the width of data item of attributes in the relation. The remaining problem is to find a routing strategy of joining all remaining relations to the site where the query was initiated with a minimum total transmission cost. In SDD-1, the system takes the simplest way by sending all the data to the query initiating node and processing at that node. In order to estimate the size of the relation after join operation, we define the join reducibility associated with each pair of relation R_i and R_j be d_{ij}. The reducibility d_{ij} has the effect that the size of data at node j after the arrival of data from node i and the join operation $(R_i \ |X| \ R_j)$ between them will produce the size of data at node j equal to $s_i s_j \ d_{ij}$. Here, $0 \leq d_{ij} \leq 1$ and $d_{ij} = d_{ji}$ for each pair of i, j . These parameters s_i, c_{ij} and d_{ij} can be measured by gathering information after system is running. We call this problem the query processing problem (QP). This is a special case of the model developed in [13]. For our purpose, we formulate the problem in the following way.

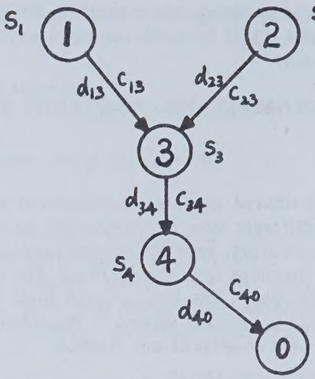
Query processing problem (QP):

Given a complete directed graph G=(V,E). Assume the size of data associated with each node i, s_i ; the unit communication cost of edge (i,j), c_{ij} ; and the join reducibility associated with edge (i,j), d_{ij} are know. We let the final node of the result of the query be v_0, and define a subtree of a tree as the usual meaning of subtree with all edges pointing to the direction of the root of the subtree. Figure 1 illustrates an example of the cost of a subtree. The objective is to find an inversely directed spanning tree toward node v_0 with minimum communication costs.

Note that this model will produce consistent estimate of the size of the result after join operations if for every permutation $\{i_{p(1)},i_{p(2)},...,i_{p(k)}\}$ of $\{i_1,i_2,...,i_k\}$, it is hold that $d_{i_1}d_{i_2}....d_{i_k} = d_{i_{p(1)}}d_{i_{p(2)}}....d_{i_{p(k)}}$. We will show that under this model, three problems of finding a routing strategy of sending required data to the desired site are NP-complete.

2.2 Complexity Theory

Recent developments in the theory of computational complexity provide a powerful method for comparing the computational difficulties of different problems. It often can provide



The Cost of This Subtree is $s_1c_{13} + s_2c_{23} + (s_1s_2s_3d_{13}d_{23})c_{34} + (s_1s_2s_3d_{13}d_{23}s_4d_{34})c_{40}$.

Figure 1: The Cost of A Subtree

information useful to algorithm designers. The application of this theory to combinational problems has aroused the interest of many researchers. The foundations for the theory are in the paper of Cook [6] and of Karp [15] who first explored the relation between the classes P and NP of (language recognition) problems solvable by deterministic and non-deterministic turing machines, respectively, in a number of steps bounded by a polynomial in the length of the input. In this context, all problems are stated in terms of recognition problems which require yes/no answers. For the combinatorial optimization problem, we transform it into the problem of determining the existence of a solution with value at most (or at least) equal to y, for some threshold y. The class NP is very extensive. It contains several classical problems ranging from the satisfiability problem of propositional calculus to the traveling salesman problem for which, despite many lengthy and intensive efforts, no efficient algorithm is known at present.

A problem is said to be NP-complete, if, intuitively, it is as hard as any problem in NP. Proving that a given problem is NP-complete typically requires two steps:

1. Showing that this problem is in NP by describing an efficient nondeterministic algorithm solving it.
2. Showing how a known NP-complete problem can be reduced to the given problem via a polynomially time-bounded transformation.

The following theorem lists those NP-complete problems that will be used in this paper to establish NP-completeness of the distributed query processing problem.

Theorem 1 : The following problems are NP-complete:

(1) Satisfiability problem in the conjunctive normal form (SAT)

Given a set $\{x_1,x_2,....,x_n\}$ of variables and a set $c_1,c_2,...,c_m$ of clauses. Each clause is the disjunction of literals (i.e. variables or negations of variables). We are asked to determine whether or not the conjunction of $c_1,c_2,...,c_m$ is satisfiable, i.e., whether there is an assignment of the values true and false to each of the variables, so that each clause contains at least one true literal.

(2) Bounded component spanning forest (BC)

Given a complete graph G=(V,E), and given a nonnegative weight $W(v)\epsilon Z_0^+$ associate with each $v\epsilon V$.

Assume positive integers $K \leq |V|$ and B are also given. Can the vertices in V be partitioned into k < K disjoint sets $V_1, V_2, \ldots, V_k$ so that for $1 \leq i \leq k$, the subgraph of G induced by V_i is connected and the sum of the weights of the vertices in V_i does not exceed B?

(3) Exact cover by 3-sets (X3C)

Given a finite set $X = \{\tau_1, \tau_2, \ldots, \tau_{3q}\}$ with $|X| = 3q$ and a collection $C = \{\sigma_1, \sigma_2, \ldots, \sigma_r\}$ of 3-element subsets of X . Does C contain an exact cover for X ? i.e., Does there exist a subcollection C' $\subseteq$ C so that every element of X occurs in exactly one member of C' ?

Proof:

(1) see reference [6]
(2) see reference [9]
(3) see reference [15]

3. COMPUTATION COMPLEXITY OF QUERY PROCESSING PROBLEM

In this section, we show that three problems of finding a routing strategy of sending required data to the desired site are NP-complete. The first problem is the query processing problem with the constraint that the number of nodes in each subtree is bounded. The second problem is the query processing problem with the constraint that the number of subtrees of the root of the directed spanning tree is bounded. The third problem consider the maximun benefit of routing strategy which has nothing to do with the minimum cost query processing problem.

3.1 Query Processing Problem With a Bounded Number of Nodes in Each Subtree (QPBS)

We formulate the problem as follows:

Assume we are given:

- a complete graph G=(V,E)
- s_i, the size of data associated with each node v_i
- c_{ij}, the unit transmission cost associated with edge (i,j)
- d_{ij}, the reducibility associated with each pair of nodes (i,j)
- B is rational constant and K is integer constant.
- v_0, the final node (query answer node)

Does there exist a directed spanning tree toward v_0 so that the cost of the spanning tree is at most B and the number of nodes in each subtree of v_0 is at most K?

Theorem 2 : QPBS is NP-complete.

Proof:

Consider the SAT problem in theorem 1. We will show that SAT is reducible to QPBS, i.e. for any instance of SAT an instance of QPBS can be constructed in polynomial-bounded time so that solving the instance of QPBS solves the instance of SAT as well. The theorem then follows from the NP-completeness of SAT in theorem 1 and the fact that QPBS belongs to NP, since any feasible spanning tree toward node v_0 can be recognized as such in polynomial time.

Given any instance of SAT, we write a set of variables as $\{x_1, x_2, \ldots, x_n\}$ and a set of clauses as $\{c_1, c_2, \ldots, c_m\}$ and define an instance of QPBS as follows:

G=(V,E) is a complete graph such that:

- $V = \{v_0, x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n, c_1, \ldots, c_m, d_1, \ldots, d_m\}$
- each node is associated with a size 2, i.e. $s_i = 2$, ∀ i
- each edge is associated with a reducibility = 1/2
- the cost of each edge is defined as follows:

$$cost(d_j, d_1) = 1, \ \forall \ j = 2, \ldots, m$$

SAT : The Boolean Formula is B = $(\bar{x}_1 + \bar{x}_2)^*(x_2 + x_3)^*(x_1 + x_2 + \bar{x}_3)$

QPBS :

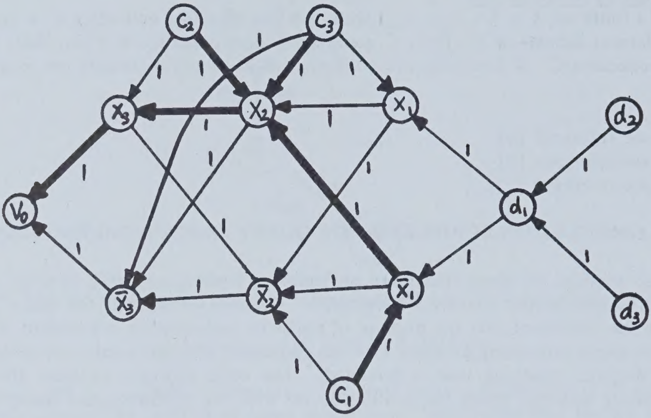


Figure 2: Illustration of Reduction from SAT to QPBS

$cost(d_1,x_1) = cost(d_1,\bar{x}_1) = 1$
$cost(x_i,x_{i+1}) = cost(x_i,\bar{x}_{i+1}) = cost(\bar{x}_i,x_{i+1}) = cost(\bar{x}_i,\bar{x}_{i+j}) = 1$
$\forall$ i = 1, 2, ... , n-1.
$cost(x_n,v_0) = 1$, $cost(\bar{x}_n,v_0) = 1$
$cost(c_i,\sigma_j) = 1$ if σ_j is a literal in c_i where σ_j is either x_i or $\bar{x}_j$.

- * all other edges have high cost.
- * B = 2 (2n+2m)
- * K = n+m

Figure 2 illustrates this reduction. Each node is associated with a size 2 and each edge has a reducibility of 1/2 . Edges of infinite length are not shown. The solution of QPBS corresponding to the truth assignment x_1 = *false*, $x_2 = x_3$ = *true* is shown with a heavy line. We claim that SAT has a solution if and only if G =(V,E) contains a spanning tree towards v_0 so that the cost of each subtree is bounded by B and the number of nodes in each subtree of v_0 is bounded by K.

(=>) Assume there is a truth assignment t satisfying SAT. Then the tree T with $d_2,d_3,...,d_m$. through d_1 and then through those 0-value literals of $\{x_i,\bar{x}_i\}$ and then to v_0 as one subtree, and $c_1,c_2,...,c_m$ through the 1-value literals and then to v_0 as another subtree of v_0 will have cost 2 (2n+2m) and each subtree has n+m nodes.

(<=) Suppose QPBS has a tree solution T. Note that T must have 2n+2m edges. Since B =2(2n+2m), and the shortest edge of G has length 1, it follows that T must consist of 2n+2m edges of length 1. Also, since there are two edges of length 1 incident upon node v_0 and K is half the number of nodes in G, T has exactly 2 subtrees rooted at node v_0, each of cardinality (not including v_0) n+m. The m nodes $d_1,d_2,...,d_m$. are obviously in one such subtree. The only way that these nodes can be connected to x_i or $\bar{x}_i$ is by a path P traversing one node from each pair $\{x_i,\bar{x}_i\}$

Let a literal have the value false if the corresponding node is in P. Since $d_1,d_2,...,d_m$ together with P exhausts the m+n vertices allowed in a subtree, the nodes $\{x_i,\bar{x}_i\}$ not in P and $\{c_j\}$ must constitute the other branch. Also, for each c_j, there is a node not in P, such that c_j is adjacent to it . Hence each clause of SAT contains a literal that is assigned the value true by the above truth assignment. Thus SAT is satisfiable.

3.2 Query Processing With a Bounded Number of Subtrees (QPBT)

We next consider the objective to be finding a directed spanning tree toward final node v_0 with a minimum number of subtrees such that the cost of each subtree is bounded by B. We prove that this problem is still NP-complete.

This problem is formulated as follows:

Assume we are given

- G = (V,E) a complete graph
- s_i, the size of data associated with each v_i
- c_i, the unit transmission cost associated with edge (i, j)
- d_{ij}, the reducibility associated with each pair of nodes (i, j)
- B and K, constants
- the final node, v_0

Does there exist a spanning tree toward node v_0 such that the cost of each subtree is bounded by B and the number of the subtrees is no greater than K?

Theorem 3 : QPBT is NP-complete.

Proof:

Note that the proof of theorem 2 works trivially for theorem 3 also. Here we give another proof. Consider the BC problem of Theorem 1. We will show that BC is reducible to QPBT, i.e., that for any instance of BC an instance of QPBT can be constructed in polynomial-bounded time so that solving the instance of QPBT solves the instance of BC as well. The theorem then follows from the NP-completeness of BC in theorem 1 and the fact that QPBT belongs to NP, since any feasible spanning tree toward node v_0 can be recognized as such in polynomial time.

Given any instance of BC, we denote the graph by $G_B = (V_B,E_B)$, and the weight of each node $v_i \epsilon V_B$ by $W(v_i) \epsilon Z_0^+$. We construct an instance of QPBT as follows:

- G =(V,E) is a complete graph with $V = V_B \cup \{v_0$ (final node)$\}$
- $s_i = 1 \ \forall$ i
- $c_{ij} = W(v_i) \ \forall$ i
- $d_{ij} = 1 \ \forall$ i, j
- K & B are the same as in BC

The subtree example in figure 3 has the communication cost $W(v_1) + W(v_2) + W(v_3) + W(v_4)$.

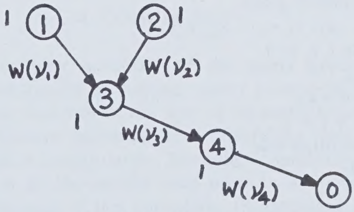


Figure 3: Illustration of Reduction from BC tp QPBT

We claim that BC has a solution iff the constructed G =(V,E) contains a spanning tree toward v_0 so that the cost of each subtree is bounded by B and the number of the subtrees is bound by K

(=>) Assume that BC has a solution, i.e. the vertices in V can be partitioned into $k \leq K$ disjoint sets $V_1, V_2, \ldots, V_k$ so that for $1 \leq i \leq k$, the subgraph of G induced by V_i is connected, and the sum of the weights of the vertices in V_i is ; .eq le B.
We construct a subtree T_i for each set V_i and connect it to v_0 and each subtree T_i has communication cost $= \sum_{l \epsilon T_i} s_i = \sum_{l \epsilon T_i} W(v_l) \leq B$, and the number of subtrees $k \leq K$.

(<=) The reverse of the argument is the same as above.

3.3 Query Processing Problem With Maximum Benefit (QM)

In the query processing problem, if we associate each edge with the unit benefit of transmitting data through that edge instead of with the unit cost, then the problem becomes finding a maximum benefit spanning tree toward node v_0. We show that this problem is still NP-complete. Note that this problem has nothing to do with minimum cost query processing problem because we cannot transform minimum cost QP problem to this problem.

We formulate the problem as follows:
Assume we are given:
- * G = (V,E)
- * for each node v_i, associate data size, s_i
- * for each pair of nodes (i, j), associate a join reducibility d_{ij}, and a unit benefit b_{ij}
- * final node v_0
- * constant B

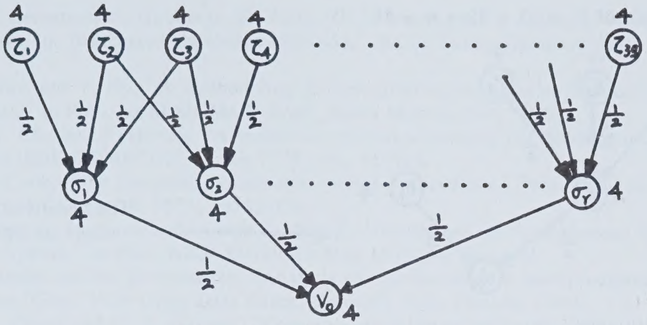
Does there exist a spanning tree toward v_0 such that the total benefit is no less than B ($\geq B$)?

Theorem 4 : QM is NP-complete.
Proof:
Considering the X3C problem in theorem 1. We will show that X3C is reducible to QM, i.e., that for any instance of X3C, an instance of QM can be constructed in polynomial-bound time such that solving the instance of QM solves the instance of X3C as well. The theorem then follows from the NP-completeness of X3C in theorem 1 and the fact that QM belongs to NP, since any feasible spanning tree toward node v_0 can be recognized as such in polynomial time.

Given any instance of X3C, we write a finite set of elements $X = \{\tau_1, \tau_2, \ldots, \tau_{3q}\}$ and a collection C of 3-element subsets of X: $C = \{\sigma_1, \sigma_2, \ldots, \sigma_r\}$, we define an instance of QM as follows:

G =(V,E) is a complete graph
- * $V = \{\sigma_1, \sigma_2, \ldots, \sigma_r, \tau_1, \tau_2, \ldots, \tau_{3q}, v_0\}$
- * for each node i, $s_i = 4$
- * $d_{ij} = 1/2 \ \forall$ i, j
- * $b(\tau_i, \sigma_j) = 1$ if $\tau_i \epsilon \sigma_j$
  $b(\sigma_j, v_0) = 1 \ \forall$ j = 1 , 2 , . . . , r
  $b(m,n) = -\infty$ otherwise
- * $B = 40q + 4r$

Figure 4 illustrates this reduction. We claim that X3C has a solution if and only if G =(V,E) contains a spanning tree with the total benefit no less than B.



Edges of - ∞ benefit are not shown

Figure 4: Illustration of reduction from X3C to QM

(=>) X3C has an exact cover C' ⊆ C, then the spanning tree with edges (τ_1,σ_j) $\forall\sigma_j \epsilon C'$ and (σ_i,v_0) i= 1,2, ..., r, has benefit
3q 4 + 32 q + 4 (r-q)
=44q - 4q + 4r
=40q + 4r > B

(<=) Considering the spanning tree toward node v_0 in fig. 4, there are only three types of first-level nodes directed toward the second-level nodes: (1) the nodes in the first level direct toward one node in the second level and leave one node in the second level without any first-level nodes directed to it, (2) two second-level nodes each have two first-level nodes directed to them, and (3) one second-level node has two first-level node directed to it and the other second level node has only one first-level node directed to it. We illustrate the three cases as in T_1,T_2, and T_3 respectively in figure 5.

As we can see from figure 4 and figure 5, we know

1. The benefit of the first level in figure 4 is constant 12q.
2. The maximun benefit of the second level in figure 4 is 32q + 4(r - q) which occur when exactly three τ_i connect to one σ_j.

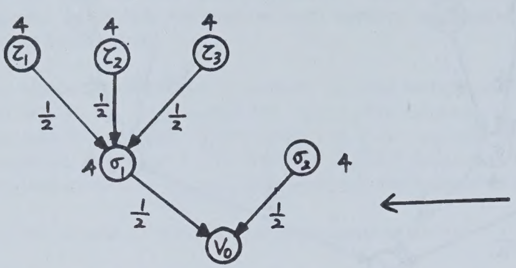
So if there exists a spanning tree toward v_0 with cost no less than B ($\geq 40q + 4r$), then this must be a spanning tree with exact three τ_i connect to one σ_j. Let $C' = \{\sigma_j \mid$ exact three τ_i connect to one σ_j in the spanning tree $\}$ Then, C' is the exact cover of X3C.

Following the same proof, if we restrict $d_{ij} = 1/2$ ∀ i, j then the problem is still NP-complete.

4. CONCLUSION

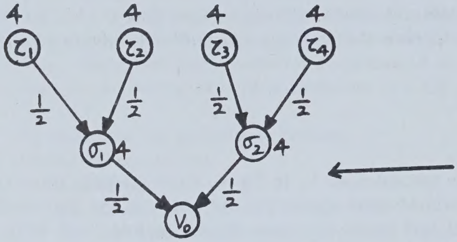
We have proven that three variations of the query processing problem (QP) are NP-complete problems. The difficulty of solving query processing problems comes from the fact that the cost of each stage depends on the cost of all previous stages. That is, we need to search over all the inversely directed spanning trees toward the final node v and compute the cost of them. The solution space is very large. The other complexity comes from the fact that at each stage (node), we want to choose the next node to move and process; we have two objectives: one is the minimization of the immediate transmission cost and the other is the maximization of the data being reduced after moving the data from one node to the other and performing the join operation between them.

T_1 : the benefit of T_1 is 12 + 32 + 4 = 48


4
4
4
ζ1
ζ2
ζ3
1/2
1/2
1/2
4
σ1
σ2
4
1/2
1/2
V0

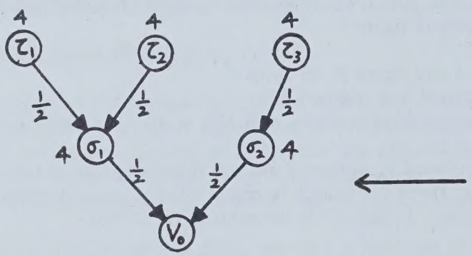
The benefit of these two edges are 32 + 4 = 36

T_2 : the benefit of T_2 is 16 + 16 + 16 = 48


4
4
4
4
ζ1
ζ2
ζ3
ζ4
1/2
1/2
1/2
1/2
σ1
4
σ2
4
1/2
1/2
V0

The benefit of these two edges are 16 + 16 = 32

T_3 : the benefit of T_3 is 12 + 16 + 8 = 36


4
4
4
ζ1
ζ2
ζ3
1/2
1/2
1/2
4
σ1
σ2
4
1/2
1/2
V0

The benefit of these two edges are 16 + 8 = 24

Figure 5: Three Types of Spanning Trees Toward Node V_0

ACKNOWLEDGEMENT

This research was support the Office of Naval Research under contract ONR/N00014-77-C-0532.

REFERENCES

[1] C. Baldissera, G. Bracchi, and S. Ceri, "A query processing stratedy for distributed data bases," in Proc. EURP-IFIP 79, 1979, pp.667-678.

[2] P. A. Bernstein and D. W. Chiu, "Using semi-join to solve relational queiries," J. Ass. Computer Mach., vol. 28, Jan. 1981.

[3] P. A. Bernstein, N. Goodman, E. Wong, C. L. Reeve, and J. Rothnie, "Query processing in a system for distributed databases (SDD-1)," ACM Trans. Database Syst., Vol. 6, Dec. 1981.

[4] D. Chiu and Y. Ho, "A methodology for interpreting tree queries into optimal semi-join expressions," in Proc. ACM SIGMOD Conf., Santa Monica, CA, 1980.

[5] W. W. Chu and P. Hurley, "A model for optimal processing for distributed databases," in proc. 18th IEEE COMPCON, Spring 1979, pp. 116-122.

[6] S. A. Cook, "The complexity of theorem-proving procedures," Proc. 3rd Ann. ACM symp. on Theory of Computing, 1971, pp151-158.

[7] R. Epstein, M. Stonebraker, and E. Wong, "Distributed query processing in a relational data base system," in Proc. ACM-SIGMOD, May 1978, pp. 169-180.

[8] R. Epstein and M. Stonebraker, "Analysis of distributed data base processing strategies," in Proc. Int. Conf. Very Large Data Bases, Montreal, P.Q., Canada, 1980.

[9] M. R. Garey and D. S. Johnson, "Computers and Intractability, A Guide to the Theory of NP-Completeness," W. H. Freeman and Company, San Francisco, 1979.

[10] N. Goodman and O. Shmueli, "Hierarchies of database state reductions," Aiken Computation Lab., Harvard Univ., Cambridge, MA, TR-18-80, 1980; Also, ACM Trans. Database Syst., to be published.

[11] A. R. Hevner and S. B. Yao, "Query processing in distributed database systems," in Proc. 3rd Berkeley Workshop on Distributed Data Management and Computer Networks, Berkeley, CA, 1978.

[12] A. R. Hevner and S. B. Yao, "Processing in distributed database systems," IEEE Trans. Software Eng., Vol. SE-5, pp.177-187, May 1979.

[13] K. T. Huang, "Query Optimization in Distributed Databases," Lab. for Information and Decision Systems, MIT, LIDS-TH-1247, Oct. 1982.

[14] R. M. Karp, "Reducibility among combinatorial problems," in R. E. Miller and J. W. Thatcher (eds.), Complexity and Computer Computations, Plenum Press, New York, pp. 85-103.

[15] C. H. Papadimitriou "The complexity of the capacitated tree problemm," Network, Vol. 8, 1978, pp. 217-230.

[16] N. G. Toan, "A unified method for query decomposition and shared information updating in distributed systems," in 1st Int. Conf. on Dist. Comp. Syst., Hunstville, Oct. 1979.

[17] N. G. Toan, "Decentralized dynamic query decomposition for distributed database systems," in Proc. ACM Pacific Conf., San Francisco, CA, 1980.

[18] G. Pelagatti and F. A. Schreiber, "Evaluation of transmission requirements in distributed database access," in Proc. ACM SIGMOD Conf., Boston, MA, 1979.

[19] M. Stonebraker and E. Neuhold, "A distributed version of INGRES," in Proc. 2nd Berkeley Workshop Distributed Data Management and Computer Networks, May 1979, pp19-36.

[20] E. Wong and K. Youssefi, "Decomposition - A strategy for query processing," ACM Trans. Database System, Vol. I. Sept. 1976.

[21] E. Wong, "Retrieving despersed data from SDD-1: A system for distributed databases," in Proc. 2nd Berkeley Workshop Distributed Data Management and Computer Networks, May 1977, pp. 217-235.

AN EVALUATION MODEL FOR INTERCONNECTION NETWORKS OF LARGE DISTRIBUTED PROCESSING SYSTEMS

G. CIOFFI(*), M. LUCERTINI(**)
(*) Dipartimento di Informatica e Sistemistica dell'Università di Roma
Via Eudossiana 18 -00184-ROMA. On leave at IBM T.J.Watson
Research Center, Yorktown Heights,NY 10598 (U.S.A.)
(**) Istituto di Analisi dei Sistemi ed Informatica del CNR e
Dipartimento di Informatica e Sistemistica dell'Università di Roma
Viale Manzoni 30
00185 - ROMA ITALY

ABSTRACT

The performance evaluation of interconnection structures for large multi-microcomputer systems is a problem investigated by many authors with different approaches.

In this work we will introduce a theoretical model especially suited to evaluate uniform and symmetric architectures realizable as networks of nodes each one implemented on one or few VLSI chips.

The model can also be applied for the evaluation of VLSI architectures although in this case some other key factors must be considered. In fact the VLSI architectures imply different problems and the cost index should take into account the peculiar interconnection and transmission limitations of VLSI realization.

We present a new proposal of interconnection structure which seems attractive for realizing networks of vast number of microcomputers with minimum ratio between busses and nodes. A comparison among several uniform and symmetric interconnection structures is carried on the ground of the evaluation model proposed.

Index terms: communication structures, distributed computers, multi-microcomputer architectures, evaluation models, routing algorithms, computer networks, topological optimization.

1. INTRODUCTION

The dramatic evolution and diffusion of VLSI technology has focused the interests of many researchers on ways to exploit large networks of computing nodes where each node can be implemented on one or few VLSI chips.

In this framework, the crucial point is to define a methodology for the design and the evaluation of the interconnection network. The evolution of the research in this field follows two main directions: systolic structures devoted to the parallel implementation of specific algorithms and distributed computer networks for general purpose applications. In this paper the second direction will be followed. A number of proposals for such general purpose systems has appeared in the literature in the last few years [1,2,3,4,5,6,7].

The common characteristics of such computer networks are:

- each node performs computation and message switching
- links/busses of the interconnection structure carry pure messages (shared memories are not allowed)
- the operating system kernel is fully distributed.

The interconnection structures fall in two different classes:

- uniform and symmetric structures (USS)
- non uniform and/or asymmetric structures.

A USS is an interconnection structure satisfying the following requirements:

i) the number of links/busses connected to each node is constant
ii) the number of nodes connected to each bus is constant
iii) the number of nodes reachable from a given node utilizing a given number of links/busses is constant.

The three requirements are not independent, for example ii) and iii) imply i).

Remark that, by this definition some architectures are not USS, like for example all tree based architectures.

The goals of this paper are twofold. In the first part some general properties of interconnection structures are pointed out and a theoretical model for the evaluation of interconnection structures is presented. The model is particularly suited for USS, although an extension to interconnection structures based on switching nodes instead of busses is briefly shown in the last section.

In the second part a new proposal of USS is presented and it is compared with several other proposals on the ground of the model.

2. A MODEL FOR THE EVALUATION OF INTERCONNECTION STRUCTURES

2.1. Key factors

Let be given N processors (nodes) to be connected one another by means of L links or busses. K(i) is the number of poles of the switch attached to processor P_i for connecting P_i to K(i) links or busses - in the most general case K(i) is function of i, whereas in uniform and symmetric structures (USS) K is constant.

For bus connected structures let W(j) be the width of bus j, defined as the number of nodes connected to the bus. The link connected structures can be considered a particular case of bus connected structures (W = 2) for most of the considerations developed in the sequel.

Unless explicitely specified, the results presented hold for both type of structures.

It is easy to verify that the following general relation holds:

$$\sum_{j=1}^{L} W(j) = \sum_{i=1}^{N} K(i) \tag{1}$$

In particular if K(i) = K and W(j) = W, as it is true for USS, the (1) becomes:

$$LW = NK \tag{1'}$$

Let R(s,i) be the number of nodes rechable in s steps with a shortest path from node i: the average number of steps S(i) in message routing, defined as the average number of busses utilized by a message from the source i to the destination, is given by:

$$S(i) = \frac{\sum_{s=1}^{s_M(i)} s \cdot R(s,i)}{N - 1} \qquad (2)$$

where $s_M(i)$ is the maximum depth of the tree resulting by means of a breath-first-search algorithm applied to the interconnection graph rooted in node i. The structure average delay S is therefore:

$$S = \frac{\sum_{i=1}^{N} S(i)}{N} \qquad (3)$$

For USS R(s,i) is independent from i, hence:

$$S = \frac{\sum_{s=1}^{s_M} s \cdot R(s)}{N - 1} \qquad (3')$$

S is a crucial parameter for the performance of the overall structure. In fact S influences the average number of messages per unit time (average traffic density) on busses and nodes as well as the corresponding average delay in message passing. The considerations in the sequel are developed under the following assumptions:

- the traffic is uniformly distributed, i.e. for every node pair the communication probability is the same
- the traffic generated by every node is the same and considered unitary.

Let $T_B(i,h)$ be the traffic density for message passing on bus h ($h = 1,\ldots,k(i)$) connected to node i(1); the average traffic density (overhead traffic) on node i is:

$$T_N(i) = \sum_{h \in \beta(i)} \frac{T_B(i,h)}{W(h)} - 1 \qquad (4)$$

where $\beta(i)$ is the set of busses connected to node i ($|\beta(i)| = k(i)$). For USS, $T_B(i,h) = T_B = NS/L$, hence:

$$\begin{cases} T_B = \dfrac{WS}{K} \\ T_N = \dfrac{K}{W} T_B - 1 = S - 1 \end{cases} \qquad (5)$$

It is worthwhile noticing that T_B and T_N depends through S on the network topology and size. Moreover the ratio between T_B and T_N+1 (global traffic on each node) is:

$$\frac{T_B}{T_N + 1} = \frac{N}{L} \qquad (6)$$

i.e. for structures with $N > L$ (typically bus connected structures) the bottlenecks are the busses.

The average delay time Δ_B at a node for accessing a bus, under standard conditions (Poisson arrivals at each node, indipendence assumptions [8,9]) is:

$$\Delta_B = \frac{1}{T_M - T_B}$$

where T_M = bus capacity expressed as number of messages per unit time.

The average end-to-end delay Δ is then:

---

(1) For sake of simplicity we have used the notation $T_B(i,h)$ to identify explicitely the bus; this does not mean that T_B depends only on i and h.

$$\Delta = S\Delta_B \tag{7}$$

In the analysis of asymptotic behavior of interconnection structures, i.e. increasing number of nodes, for almost all the structures of practical interest, T_B is a monothonic increasing function of N. Therefore it is meaningless to consider T_M constant; a reasonable assumption consists on supposing constant the average delay time on a bus: $\Delta_B = \alpha_1$ (2). With this assumption the end-to-end delay becomes:

$$\Delta = \alpha_1 S \tag{8}$$

and the bus capacity:

$$T_M = T_B + \frac{1}{\alpha_1} \tag{9}$$

2.2. The cost index

A cost index suitable for comparing different architectures will be defined as the product of two terms:

I = hardware cost per node x message average time delay.

The smaller is the index I, the better is the tradeoff between structural complexity of the network and its performance. The average time delay is given by the (8). The hardware cost per node, in the general case, can be considered the sum of four cost terms:

i) node-bus interface channels: C_c
ii) busses: C_B
iii) nodes: C_N
iv) bus arbiters: C_A

C_c is assumed to be proportional to the number of channel interface,time the width of the busses:

$$C_c = \alpha_2 \sum_{i=1}^{N} \sum_{j\in\beta(i)} W(j) \tag{10}$$

In fact, for each interface channel,the transmission speed must be proportional to the number of nodes connected to the same bus.

The cost of a bus j is assumed to be proportional to its capacity $T_M(j)$, hence:

$$C_B = \alpha_3 \sum_{j=1}^{L} T_M(j) = \alpha_3 \sum_{j=1}^{L} T_B(j) + \alpha_4 L \tag{11}$$

With similar considerations we obtain:

$$C_N = \alpha_5 \sum_{i=1}^{N} (T_N(i) + 1) \tag{12}$$

$$C_A = \alpha_6 \sum_{j=1}^{L} W(j) \tag{13}$$

For USS the (10), (11), (12) and (13) simplify in:

$$C_c = \alpha_2 NKW \tag{10'}$$

$$C_B = \alpha_3 T_B L + \alpha_4 L \tag{11'}$$

$$C_N = \alpha_5 NS \tag{12'}$$

$$C_A = \alpha_6 LW \tag{13'}$$

---

(2) In the sequel all constants will be indicated by α_i $(i = 1,2,\dots)$.

Finally the index I results:

$$I = \frac{\Delta}{N}(C_C + C_B + C_N + C_A) \tag{14}$$

and for USS:

$$I = \alpha_1 S(\alpha_2 KW + \alpha_3 S + \alpha_4 \frac{K}{4W} + \alpha_5 S + \alpha_6 K) \tag{14'}$$

Asymptotically we obtain:

$$I = O(S(KW + S)) \tag{15}$$

2.3. Lower bounds for the cost index of USS

In the previous paragraph we have obtained a cost index depending only on K,W,S. In order to evaluate different architectures it can be useful to compare the cost index behavior against a theoretical lower bound. It is common, in order to simplify the formulae, to consider the asymptotical behavior of the cost index as function of N. Therefore we must express W,K and S as function of N, and this will be done for the specific architectures analyzed in the next section. As far as the lower bound is concerned, we can derive a lower bound for I starting from a lower bound for S, as function of N,W and K.

As stated by (15), the cost index depends on S and $K \times W$. Now, if $k = 1$ the only interconnecting structure is the single bus, and since for this structure $W = N$ and $S = 1$, the cost index becomes $I = O(N)$. If $K = 2$ and $W = 2$, the only interconnecting structure is the linear array, for which $S = N/4$ and $I = O(N)$. On the other hand, if $S = 1$ the interconnecting structure must have either $W = N$ and $K \geq 1$ or $W = 2$ and $K = N-1$; in both cases $I = O(N)$. Therefore in order to obtain a cost index better than $O(N)$ it is necessary that $S > L$, $K \geq 2$, $W \geq 2$, and $K \times W \geq 6$.

THEOREM 1: $S = \Omega\left(\frac{\log N}{\log W + \log K}\right)$.

PROOF. Consider the maximum number of nodes reachable in p steps for given values of K and W; R(p) is bounded by:

$$R(p) \leq K(W-1)[(K-1)(W-1)]^{p-1} \tag{16}$$

Let p_{max} be the maximum number of steps needed to reach all the nodes if the strict inequality in the (16) holds for at least a p, and $\bar{p}$ the same if the (16) holds with the equality for all p except possibly $\bar{p}$: $p_{max} \geq \bar{p}$.

Since $S = \sum_{p=1}^{p_{max}} p\, R(p)/(N-1)$, substituting the right-hand side of the (16) to R(p) this implies an increase of the number of nodes multiplied for smaller p, hence

$$S > \frac{K(W-1)}{N-1} \sum_{p=1}^{\bar{p}} p\gamma^{p-1} \qquad \text{where} \quad \gamma = (K-1)(W-1)$$

Since $N-1 \leq \sum_{p=1}^{\bar{p}} K(W-1)\gamma^{p-1}$ we can state:

$$S > \frac{\sum_{p=1}^{\bar{p}} p\gamma^{p-1}}{\sum_{p=1}^{\bar{p}} \gamma^{p-1}} = \frac{\frac{d}{d\gamma}[\gamma \sum_{p=0}^{\bar{p}-1} \gamma^p]}{\sum_{p=0}^{\bar{p}-1} \gamma^p} = \bar{p}\left(\frac{\gamma^{\bar{p}}}{\gamma^{\bar{p}}-1} - \frac{1}{\bar{p}(\gamma-1)}\right) \geq c\bar{p}$$

where $\frac{1}{2} \leq c \leq 1$ and $c \to 1$ for increasing values of $\bar{p}$ and/or γ. On the other hand, with the hypothesis on K and W ($KW \geq 6$), $N \leq 4\gamma^{\bar{p}}$, hence:

$$\bar{p} \geq \frac{\log N - 2\log 2}{\log \gamma} \quad \text{and} \quad S \geq \frac{\log N}{\log \gamma}$$

Substituting in the (15) the expression $\log N/\log\gamma$ instead of S, we obtain a lower bound for I as function of N and γ:

$$I = \Omega\left[\left(\frac{\log N}{\log \gamma}\right)^2 + \gamma\frac{\log N}{\log \gamma}\right] \qquad (15')$$

For $2 \leq \gamma \leq N$ the two components of the right-hand side of the (15') vary as shown in fig. 1.

Remark that γ = constant implies:

$$I = \Omega(\log^2 N)$$

For $\gamma = f(N)$ we can derive a different lower bound. In fact for $\gamma < \bar{\gamma}$ the first component of the (15') dominates the second one, whereas for $\gamma > \bar{\gamma}$ the second one is dominant. For $\gamma = \bar{\gamma}$ the two components are equal, therefore $\bar{\gamma}$ minimize the order of the lower bound of I. On the other hand $\bar{\gamma}$ depends on N as $\bar{\gamma}^{\bar{\gamma}} = N$, and since $0.47\frac{\log N}{\log\log N} < \bar{\gamma} < 1.37\frac{\log N}{\log\log N}$, $\bar{\gamma} = \Theta\left[\frac{\log N}{\log\log N}\right]$. Therefore we have proved the following:

THEOREM 2: $I = \Omega\left[\left(\frac{\log N}{\log\log N}\right)^2\right]$.

3. D-DIMENSIONAL ARRAY: A NEW PROPOSAL

3.1. The topology

On the basis of the cost index derived in the previos section, new USS architectures have been investigated. In this section a new proposal will be described with the asymptotical behavior reaching the cost index lower bound introduced in theorem 2.

The D-Dimensional Array (DDA) proposed in this section can be considered a D-Dimensional W-Wide Spanning Bus Hypercube (SBH) [1] where each node is substituted with a local bus of width D and each of the D nodes of the local bus is also connected to one of the D spanning busses, each spanning a different dimension in the hypercube address space. A DDA with D = 2 is shown in fig. 2.

Since in a SBH with $W = D$, $N_{SBH} = D^D$, in the DDA $N_{DDA} = D^{D+1}$. Moreover in the SBH $K = D$, whereas in the DDA $K = 2$. The number of busses L is immediately derived from (1'): $L_{DDA} = 2\cdot D^D$, $L_{SBH} = D^D$. From the point of view of physical feasibility a basic requirement is the capability to expand the network without increasing K and W beyond relatively small bounds due to bus bandwidth, driver/receiver characteristics, physical limits on the number of channels per node.

In the DDA $K = 2$ and W is function of N, however small values of W, i.e. D, produce vast number of nodes. For example $D = 5$ implies $N = 15.625$, $D = 10$ implies $N = 10^{11}$. Remark that in the SBH as well as in the DDA the width W_S of the spanning busses can be different from D. In this more general case, the whole number of nodes is:

$$N_{SBH} = W_S^D; \quad N_{DDA} = D\cdot W_S^D$$

Obviously the DDA, if $W_S \neq D$, is no more a USS, since spanning busses and local busses have different width and traffic. However the USS results can be utilized introducing the average values of bus width and traffic.

The facility to vary W_S independently from D is in practice very useful to vary in a finer way the number of nodes.

An efficient way for naming nodes and busses, in order to simplify routing and interconnection problems is the following.

The nodes are named with a vector $n = \langle n(0), n(1), \ldots, n(D) \rangle$ where each $n(i)$ is an integer: $1 \leq n(i) \leq D$. The first entry, $n(0)$, indicates the position on the local bus, the remaining entries indicate the coordinates in the hypercube address space. In the same way a vector $\ell = \langle \ell(1), \ell(2), \ldots, \ell(D) \rangle$ indicates a local bus, and a vector $s = \langle s(1), \ldots, s(i-1), x, s(i+1), \ldots \ldots, s(D) \rangle$ indicates a spanning bus, where the symbol x (don't care) in the i-th position denotes the spanned dimension. Since each node is connected to an L-BUS and an S-BUS, the names of the two busses connected to node N are:

L-BUS: $\langle n(1), \ldots, n(D) \rangle$; S-BUS: $\langle n(1), \ldots, n(n(0)-1), x, n(n(0)+1), \ldots, n(D) \rangle$.

For example in a 4-DA the node $\langle 3,4,3,2,1 \rangle$ is connected to the L-BUS $\langle 4,3,2,1 \rangle$ and to the S-BUS $\langle 4,3,x,1 \rangle$.

Note that the L-BUS $\langle 4,3,2,1 \rangle$ is connected to all the nodes $\langle x,4,3,2,1 \rangle$, whereas the S-BUS $\langle 4,3,x,1 \rangle$ is connected to all nodes $\langle 3,4,3,x,1 \rangle$ $(x = 1,2,3,4)$.

3.2. The routing

The routing algorithm to send a message from a source n_s to a destination n_d via a shortest path is straightforward. In fact at each step the message must be transmitted alternatively via a spanning bus or a local bus. We will introduce the routing by means of an example and successively the formal structure of the algorithm will be presented.

Let $D = 4$, $n_s = \langle 3,1,4,2,3 \rangle$, $n_d = \langle 1,2,1,3,4 \rangle$.

Since $n_s(0) = 3$ the message will be sent via S-BUS $\langle 1,4,x,3 \rangle$ to $n_1 = \langle 3,1,4,3,3 \rangle$ so that the coordinate in the third dimension is adjusted to 3. The node n_1 sends the message via L-BUS $\langle 1,4,3,3 \rangle$ to $n_2 = \langle 4,1,4,3,3 \rangle$ in order to select the next spanning bus. The process is repeated untill all coordinates match the n_d ones. The complete sequence of nodes is therefore:

$$
\begin{aligned}
n_s = \langle 3,1,4,2,3 \rangle &\xrightarrow{S} \langle 3,1,4,3,3 \rangle \xrightarrow{L} \langle 4,1,4,3,3 \rangle \xrightarrow{S} \langle 4,1,4,3,4 \rangle \xrightarrow{L} \\
&\xrightarrow{L} \langle 1,1,4,3,4 \rangle \xrightarrow{S} \langle 1,2,4,3,4 \rangle \xrightarrow{L} \langle 2,2,4,3,4 \rangle \xrightarrow{S} \langle 2,2,1,3,4 \rangle \xrightarrow{L} \\
&\xrightarrow{L} \langle 1,2,1,3,4 \rangle = n_d
\end{aligned}
$$

The routing algorithm performed by each node n_i belonging to the path $n_s = n_o, n_1, \ldots, n_p = n_d$ can be described by the following:

<u>Transmitting procedure for node n_i $(i = 0,1,\ldots,p)$</u>

```
BEGIN
  K:=1 : c:=$n_i(0)$
  IF $n_i(c) \neq n_d(c)$ THEN
     Transmit via S-BUS to $n_{i+1} = \langle n_i(0), \ldots, n_d(c), \ldots, n_i(D) \rangle$
  ELSE
      WHILE $K \leq D$ and $n_i(c) = n_d(c)$ DO
             K:=K+1; C:=C+1
             IF C=D+1 THEN C:=1
      END WHILE
     IF $K \leq D$ THEN
        Transmit via L-BUS to $n_{i+1} = \langle c, n_i(1), \ldots, n_i(D) \rangle$
     ELSE IF $n_i(C) \neq n_d(C)$ THEN
        Transmit via L-BUS to $n_d$
     ELSE STOP
END
```

This algorithm assumes that each node transmits to a receiving node, via an L-BUS or an S-BUS, the address of n_d only. The maximum number of nodes along the path is 2D, and the complexity of the routing algorithm for the whole path is $O(D^2)$.

This algorithm can be improved appending at the transmitted message the number of modified coordinates, such that the complexity of the routing algorithm for the whole path is O(D). Moreover the maximum number of nodes visited along the path can be diminished by one if the last coordinate to modify is that corresponding to $n_d(\emptyset)$. In the above example $n_d(\emptyset) = 1$, therefore the coordinate in the first direction is left as the last to modify. The sequence of nodes becomes:

$$n_s = \langle 3,1,4,2,3\rangle \xrightarrow{S} \langle 3,1,4,3,3\rangle \xrightarrow{L} \langle 4,1,4,3,3\rangle \xrightarrow{S} \langle 4,1,4,3,4\rangle \xrightarrow{L} \langle 2,1,4,3,4\rangle \xrightarrow{S} \langle 2,1,1,3,4\rangle \xrightarrow{L} \langle 1,1,1,3,4\rangle \xrightarrow{S} \langle 1,2,1,3,4\rangle = n_d$$

Optimized transmitting procedure for node n_i $(i = 0,1,\dots,p)$

```
BEGIN
 INPUT: $K, n_d$
 IF $K = D$ THEN STOP
 ELSE $C := n_i(\emptyset)$
    IF $n_i(C) \neq n_d(C)$ THEN $K := K+1$, transmit via S-BUS
                         to $n_{i+1} = \langle n_i(\emptyset),\dots,n_d(C),\dots,n_i(D)\rangle$
    ELSE $C := C+1$
     WHILE $n_i(C) = n_d(C)$ AND $K < D$
       $K := K+1$; $C := C+1$
       IF $C = D+1$ THEN $C := 1$
       IF $C = n_d(\emptyset)$ THEN $C := C+1$
       IF $C = D+1$ THEN $C := 1$
    END WHILE
 IF $K < D$ THEN transmit via L-BUS to $n = \langle C, n_i(1),\dots,n_i(D)\rangle$
 ELSE transmit via L-BUS to $n_d$
END
```

3.3. The cost index

Since K = 2 and W = D, in order to evaluate I it is necessary to calculate S (structure average delay).

Remark that the routing introduced in the previos paragraph utilizes a maximum number of 2D-1 busses. In a spanning bus hypercube the average path lenght (S_{SBH}) is:

$$S_{SBH} = \frac{\sum_{p=1}^{D} p\binom{D}{p}(D-1)^p}{D^D-1} = (D-1)\left(1+\frac{1}{D^D-1}\right) \simeq D-1 \tag{18}$$

S_{DDA} can be derived from S_{SBH} with the assumption that the unitary step performed in the SBH is splitted in two sub-steps, along a L-BUS and an S-BUS respectively. Therefore at each step the number of reachable nodes is D times the number of nodes reachable in the SBH, but with a double number of busses. Remark that this approximate evaluation of S_{DDA} is an upper bound for S_{DDA}.

Since this upper bound leads to a cost index value asymptotically equal to the theoretical lower bound of Th. 2, this approximation can be accepted

$$S_{DDA} \leq 2(D-1)\left(1+\frac{1}{D^{D+1}-1}\right) \simeq 2(D-1) \tag{19}$$

The cost index I_{DDA}, as function of D, is:

$$I_{DDA} = O(D^2)$$

It is more convenient to express the cost index as function of N, in order to make a comparison with other architectures.

For DDA $N = D^{D+1}$, and for $D > 2$, $\frac{\log N}{\log\log N} < D < 1.33(\frac{\log N}{\log\log N})$, hence $D = \theta(\frac{\log N}{\log\log N})$

$$I_{DDA} = O\left(\left(\frac{\log N}{\log\log N}\right)^2\right) \tag{20}$$

Therefore, under the assumptions behind Th. 2, DDA is asymptotically optimal.

3.4. Comparison with other USS

Several USS architectures have been proposed in the literature for interconnecting a large amount of VLSI chips, or in general standard computer modules. Wittie [1] have analysed some USS introducing scores for some key factors caracterizing the networks and a global index obtained by summing up the scores. The USS analysed are the following:

1) Ring: $K = 2$, $W = 2$, $S = N/2$
2) Multiple global busses: $K = L$, $W = N$, $S = 1$
3) Fully connected: $K = N-1$, $W = 2$, $S = 1$
4) Nearest neighbor mesh [10]: $K = 2D$, $W = 2$, $N = M^D$, $S = DM/4$
5) SBH [1]: $K = D$, $N = W^D$, $S = D-1$
6) Dual buss hypercube [1]: $K = 2$, $N = W^D$, $S = 2(D-1)$
7) Cube connected cycle [11]: $K = 3$, $W = 2$, $N = D2^D$, $S = 7D/4$
8) DDA: $K = 2$, $N = D \cdot W^D$, $S = 2(D-1)$

where for the structure n.° 4, M is the number of nodes per dimension.

Note that, strictly speaking, the Cube Connected Cycle is not USS since the traffic in cross-links ($T_B = D$) is different from the traffic in cycle links ($T_B = 5D/4 - 3$). The Dual Bus Hypercube and DDA are USS if $W = D$.

The cost index introduced in section 2 has been applied to the above topologies with the assumption $M = W = D$. The behavior of the cost indexes are:

$$I_1 = O(N^2);\quad I_2 = O(L \times N);\quad I_3 = O(N);\quad I_4 = O\left(\left(\frac{\log N}{\log\log N}\right)^4\right);$$

$$I_5 = O\left(\left(\frac{\log N}{\log\log N}\right)^3\right);\quad I_6 = O\left(\left(\frac{\log N}{\log\log N}\right)^2\right);\quad I_7 = O(\log^2 N);$$

$$I_8 = O\left(\left(\frac{\log N}{\log\log N}\right)^2\right)$$

From the above cost indexes we can confirm that the architectures lieing on the lattice points within a hypercube have a better behavior than the simpler ones. The Dual Bus Hypercube and the DDA reach the theoretical USS lower bound for bus interconnected structures, whereas the Cube Connected Cycle reach the lower bound for link interconnected structures.

4. FINAL CONSIDERATIONS

The analysis presented in this paper concerns mainly USS based on busses or links. However the analysis can be easily extended to USS where instead of busses there are switching nodes interconnecting computing nodes via links. The parameters L, W and T_B become in this case the number of switching nodes, the number of computing nodes connected to each switching node and the average traffic through each switching node respectively. In practice W cannot grow beyond a given phisical limit and for increasing values of W, each switching node becomes a switching tree interconnecting W computing nodes. In a VLSI framework W is bounded by 3 or 4 and the switching tree is the normal way a multiple interconnection is realized.

Remark that, if the busses are implemented as switching trees, the average number of steps in message passing increases by an average factor $O(\log W)$. Furthermore the cost C_B must be substituted by the cost C_S of the switching trees and the cost C_A must be eliminated.

The cost C_S is given by the dimension of the switching tree $(O(W \log W))$ time the number L of switching trees time the average traffic in the switching nodes $(O(W))$:

$$C_S = O(LW^2 \log W)$$

With simple algebra the DDA cost index becomes $O(\log^2 N)$, i.e. the same order of the cost index lower bound for USS with bounded K and W.

Further developments of the DDA architecture, presently in progress, concerns the specification of process sincronization and cooperation primitives, dynamic routing policies, recovery and rerouting for fault tolerant systems, and the embedding of computational graphs into the DDA.

REFERENCES

[1] WITTIE: Communication structures for large networks of microcomputers. IEEE-TC, Vol. C-30, n. 4, 1981.

[2] WU, LIU: A cluster structure as an Interconnection network for large multimicrocomputer systems.IEEE-TC, Vol. C-30, n. 4, 1981.

[3] HARRIS, SMITH: Hierarchical multiprocessor organizations. 4th Symp. on Comp. Arch., 1977.

[4] SWARTZLANDER, GILBERT: Supersystems: technology and architecture. IEEE-TC, vol. C-31, 1982.

[5] THOMPSON: Generalized conncection networks for parallel processor intercommunication. IEEE-TC, vol. C-27, 1978.

[6] SULLIVAN, BASHKOW: A large scale homogeneous, fully distributed parallel machine. 4th Symp. on Comp. Arch., 1977.

[7] WU, FENG, LIN: Star: a local network system for real-time management of imagery data. IEEE-TC, vol. C-31, 1982.

[8] KLEINROCK: Communication nets: stochastic message flow and delay. New York: Mc Graw-Hill, 1964.

[9] KERMANI, KLEINROCK: A trade off study of switching systems in computer communication networks. IEEE-TC, vol. C-29, 1980.

[10] BARNES et al.: The ILLIAC IV computer. IEEE-TC, vol: C-17, 1968.

[11] PREPARATA,VUILLEMIN: The cube-connected-cycle: a versatile network for parallel computation. 20th FOCS, 1979.

[12] METCALFE, BOGGS: Ethernet: distributed packet-switching for local computer networks. Comm. ACM, vol. 19, 1976.

[13] RILEY, BARON: Design and Evolution of a synchronous triangular interconnection scheme for interprocessor communication. IEEE-TC, vol. C-31, 1982.

[14] GONZALES, JORDAN: A framework for the quantitative evaluation of distributed computer systems. IEEE-TC, vol. C-29, 1980.

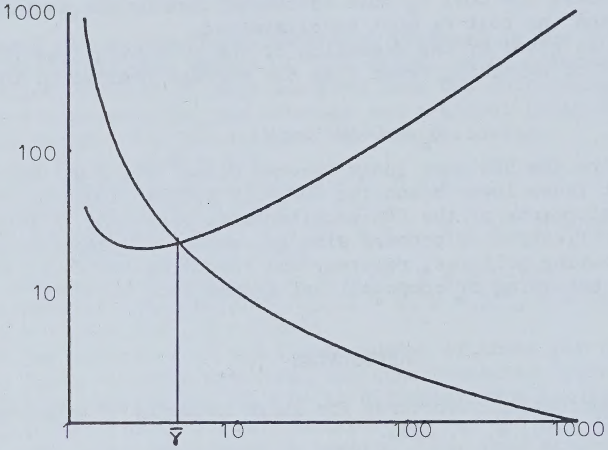


Fig.1

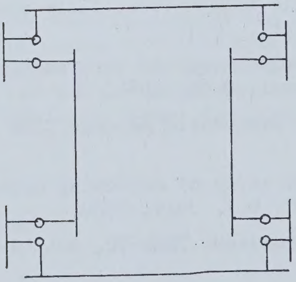


Fig.2

SORTING ON GATED-BUS ARRAY PROCESSORS

MING-YANG CHERN and TADAO MURATA

Department of EECS
Northwestern University
Evanston, IL 60201

Department of EECS
Univ. of Illinois at Chicago
Chicago, IL 60680

SUMMARY

Recently we introduced a VLSI-compatible parallel architecture called the concurrent data-loading array processor (CDLAP) and showed that the CDLAP can execute various matrix operations with high efficiency (in many cases more efficiently than systolic arrays) [1, 2]. This paper presents a sorting scheme based on gated-bus array processors --- a modification of the CDLAP with gated buses. The sorting of n data on such an array with n processing elements can be completed in n memory read cycles.

For sorting on a linear array with gated bus, each processing element (PE) in this array has an input port and an output port as shown in Fig.1. The on-bus gate G1, which is controlled by each PE, is located between the two I/O ports. This configuration is such that the output of a PE can be connected to the input of its left neighbor. The output port may or may not be connected to the bus depending on whether the gate G2 is closed or open. The bus can be used for data broadcasting, when all the on-bus gates are closed.

Without loss of generality, we assume that an ascending order is desired in this sorting. Before the sorting is started, the array must be initiated with a minimum possible value. For convenience of illustration, let 0 be this minimum possible value. An example is shown in Fig.2, where circles represent the gates which are open. In the first step, the new input value is concurrently loaded into all PEs and compared with the stored value in each PE. The result of the comparison in each PE is then used to control the gate setting and the data transfer. If the stored value is larger in a given PE, nothing will be done for the next step. Otherwise, the on-bus gate corresponding to this PE will be open and the stored value will be output to the segmented bus. Since the stored data in this array have been sorted in the previous iterations, all the PEs left to any PE with its on-bus gate open will have the same action. The status of the array at this moment is shown in Fig.2(b). Then another concurrent data-loading is performed for these PEs with their on-bus gates open. The net effect is that the data no larger than the input value are all shifted to the left by one position while the new input datum is inserted into the right position in the given array. The result after this iteration is shown in Fig.2(c).

The above sorting scheme on the linear array is simple. However, it is not appropriate for sorting a large number of data. If the data broadcast delay is 10 nsec for having 20 PEs in the linear array, then the broadcast delay for having 400 PEs would be about 200 nsec. This makes the time requirement of this sorting scheme $O(n^2)$. And this deteriorates the performance of this sorting severely even for a moderate number of data.

To solve this problem, a two-dimensional array is used. The sorting scheme is illustrated in Fig.3. The whole array is initially set to a minimum possible value which is 0 as in the previous example. In Fig.3(a),

---

This work was supported by the National Science Foundation under Grant ECS 81-05649.

the input value is broadcast to each column in exactly the same way as the previous scheme. The result is shown in Fig.3(b). The minimum value in each column enters the output bus register at the top. The contents of the bus registers are then shifted one position to the left. The right end of the bus registers is filled with the input value (i.e. 15 in the example). The data in these bus registers are then back-broadcast to the array as shown in Fig.3(c). At this stage, only the PEs in the bottom row may take actions. If their originally stored values are larger than the input value (which is 15), do nothing. Otherwise, load the back-broadcast data. And this completes the current iteration of this sorting. The result is shown in Fig.3(d).

Both sorting shcemes proposed here require the processing time proportional to the number of data elements. Let t_b represents the data broadcast time (including data loading time), t_c the time for one comparison, t_g the gate setting time, and t_s the data shift time. Then for sorting n data on the linear gated-bus array, the time requirement would be $n*(t_b + t_c + t_g + t_s)$. However, the data broadcast delay would be roughly proportional to n for a large linear array with n PEs. For sorting n^2 data on an n x n array, the required time is $n^2*(2t_b + t_c + t_g + 2t_s)$. The data broadcast time t_b is short in the latter case, because the number n (the number of PEs connecting to the same bus) can not be too large. From a practical point of view, the time required for sorting each datum on our two-dimensional array processor can possibly shorter than the memory access time. Thus this sorting can be finished as the data to be sorted are read serially from the memory.

Although some researchers have proposed other parallel sorting schemes (such as in [3, 4]) which have lower order of time requirement in their evaluation, the data loading time was mostly neglected. In our simple design, the sorting can be completed at the time the data are loaded. Therefore it can be considered to be among the most advantegeous ones.

REFERENCES

[1] M.Y. Chern and T. Murata, "Efficient Matrix Multiplications on a Concurrent Data-Loading Array Processor", Proc. 1983 Int. Conf. on Parallel Processing, pp.90-94.

[2] M.Y. Chern and T. Murata, "A Fast Algorithm for Concurrent LU Decomposition and Matrix Inversion", Proc. 1983 Int. Conf. on Parallel Processing, pp.79-86.

[3] C.D. Thompson and H.T. Kung, "Sorting on a mesh-connected parallel computer", Comm. ACM, Vol.20, No.4, 1977, pp.263-271.

[4] D. Nassimi and S. Sahni, "Bitonic sort on a mesh-connected parallel computer", IEEE Trans. Computers, Vol.27, No.1, 1979, pp.2-7.

[5] S. Todd, "Algorithm and hardware for a merge sort using multiple processors", IBM J. Res. & Devel., Vol.22, No.5, 1978, pp.509-517.

[6] F.P. Preparata, "New parallel-sorting schemes", IEEE Trans. on Computers, Vol.27, No.7, 1978, pp.669-673.

[7] H. Yasuura, N. Takagi, and S. Yajima, "The parallel enumeration sorting scheme for VLSI", IEEE Trans. on Computers, Vol.31, No.12, 1982, pp.1192-1201.

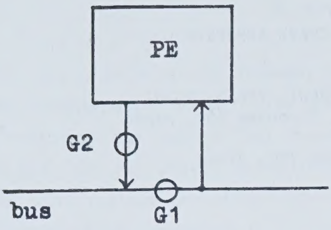


Fig.1 I/O arrangement of a PE in a gated-bus array processor

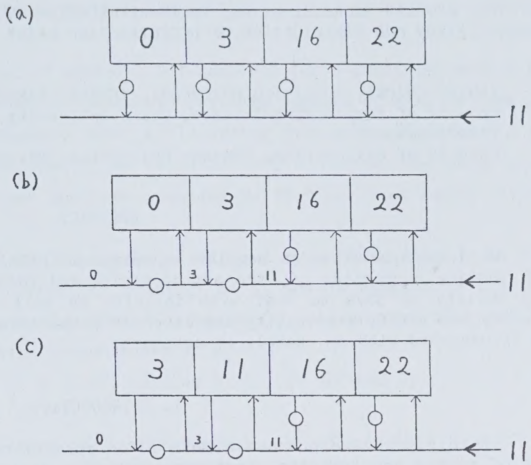

(a)
0 3 16 22
11
(b)
0 3 16 22
0 3 11
11
(c)
3 11 16 22
0 3 11
11

Fig.2 Sorting on a linear gated-bus array processor

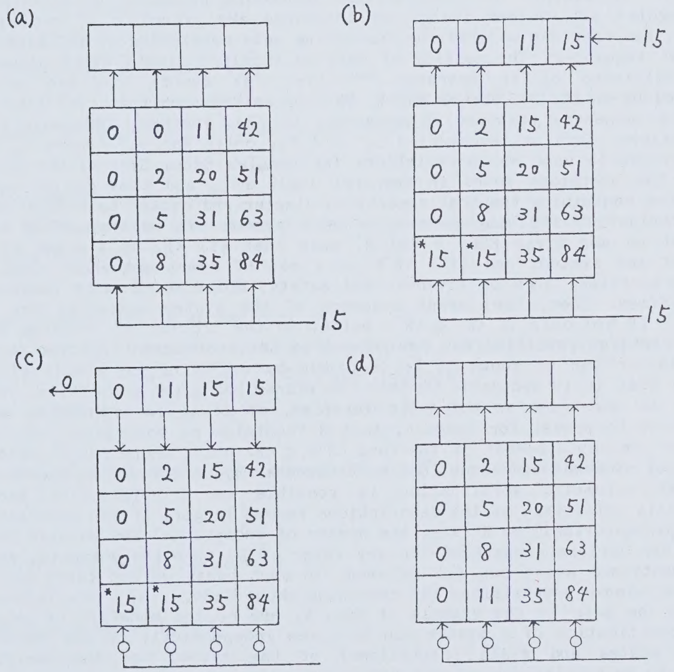

(a)
0 0 11 42
0 2 20 51
0 5 31 63
0 8 35 84
15
(b)
0 0 11 15
15
0 2 15 42
0 5 20 51
0 8 31 63
*15 *15 35 84
15
(c)
0
0 11 15 15
0 2 15 42
0 5 20 51
0 8 31 63
*15 *15 35 84
(d)
0 2 15 42
0 5 20 51
0 8 31 63
0 11 35 84

Fig.3 Sorting on a two-dimensional gated-bus array processor

EXTENDED REGULAR EXPRESSIONS AND THEIR APPLICATION TO SPECIFICATION AND VERIFICATION OF DECENTRALIZED DAISY CHAIN ARBITERS

ICHIRO SUZUKI, YOSHIHIKO MOTOHASHI, KENICHI TANIGUCHI, TADAO KASAMI
Faculty of Engineering Science, Osaka university, Toyonaka 560, Japan
TAKUJI OKAMOTO
Faculty of Engineering, Okayama University, Okayama 700, Japan

ABSTRACT

We introduce extended regular expressions (EREs) that can be used for specifying, and proving properties of, the set of finite and infinite event sequences of a system. The ability of EREs to deal with infinite as well as finite sequences allows us to specify and verify eventuality and liveness properties quite naturally. The use of EREs is illustrated with an example of arbiters.

1. INTRODUCTION

In this paper we present a method for specifying, and proving properties of, the set of finite and infinite event sequences of a system. The method is illustrated with an example of arbiters.

When proving properties of systems, it is convenient, and sometimes even necessary, that we consider event sequences of infinite length as well as finite sequences. In particular, some liveness and eventuality properties can be expressed in terms of infinite event sequences naturally (see Sec. 4 for an example of arbiters and also [3] for an example of communication protocols using an unreliable channel). Extended regular expressions (EREs) we introduce are extension of ordinary regular expressions, and they can be used for describing sets consisting of infinite as well as finite event sequences. The ability of EREs to handle infinite event sequences comes from the existence of an operator "ω" (let <R> denote the set of sequences characterized by an ERE R): For an ERE R, $(R)^{\omega}$ is an ERE such that $<(R)^{\omega}>$ is the set of all infinite sequences composed of sequences in <R>. Further, EREs have a number of useful notations, such as "eventually e_1" and "e_1 onlyafter e_2" (where e_1 and e_2 are events) that can be used as abbreviations for complex EREs. Some of the notations are similar to the operators found in temporal logic [5], and they can be used to make specifications containing temporal assertions simpler and easier to understand.

The problem of verifying properties of a system can be formulated as follows: Suppose that we are given EREs R and R´ such that (i) <R> is the set of all event sequences of the system, and (ii) <R´> is a set of event sequences that have some desirable properties, such as liveness and safety, which we wish to test whether the system satisfies. Then, "any event sequence of the system satisfies the properties given by R´" if and only if $<R> \subseteq <R'>$ holds. So the problem of deciding whether the system has certain properties can be reduced to the containment problem for two EREs. Since the latter can be shown to be decidable by extending the result of McNaughton [6], we see that it is decidable whether the system has given properties. Therefore it is possible to construct an automatic verifier, at least in principle. At the same time, in order to prove, for example, that R "contains no contradiction", we have to test whether the containement of the form $<R''> \subseteq <R>$ holds where ERE R" describes, for example, a set of event sequences that must necessarily be allowed in the system.

Although automated verification is possible in principle, it tends to be computationally infeasible as the descriptions become larger. If the description of the system is "parameterized" by K (say, the number of subsystems) and we wish to show that the system has certain properties for any value of K, then the automatic verification technique mentioned above can not be used. In such cases we can carry out proofs by using several simple propositions for reasoning about EREs.

As will be seen in the example of Sec. 4, one of the benefits of using EREs is that the specification of a system can be given independently of the inner structure (e.g., the states and state transitions) of the sytem, and the verification of properties can be carried out without knowing how the system is implemented.

The class of sets of sequences described by EREs is closely related to that of ω-regular languages whose theoretical aspects have been studied extensively [2,4,6].To the authors´ knowledge, this paper reports the first result ever published on applications of ω-regular-like languages.

2. EXTENDED REGULAR EXPRESSIONS

Let Σ be a finite nonempty set of symbols. A $*$-sequence (or ω-sequence) over Σ is any concatenation of a finite number (or countably infinite number) of symbols from Σ. Σ^* denotes the set of all $*$-sequences over Σ, including the empty sequence λ. Σ^ω denotes the set of all ω-sequences over Σ. We let $\Sigma^\infty=\Sigma^*\cup\Sigma^\omega$. ϕ denotes the empty set. (In what follows the word "sequence" when not preceded by "$*$-" nor "ω-" means either $*$-sequence or ω-sequence.)

For $\alpha\in\Sigma^*$ and $\beta\in\Sigma^\infty$, $\gamma=\alpha\beta$ denotes the sequence obtained by concatenating β to the tail of α. α (or β) is called a prefix (or postfix) of γ, and is denoted by γ/β (or $\alpha\backslash\gamma$). For $\alpha\in\Sigma^*$, $(\alpha)^\ell$ (or $(\alpha)^\omega$) denotes the $*$-sequence (or ω-sequence) obtained by cocatenating ℓ (or countably infinite) occurrences of α.

The concatenation of subsets U, V of Σ^∞, denoted by UV, is defined by

$$UV = \{\gamma \mid \gamma=\alpha\beta,\ \alpha\in U\cap\Sigma^*,\ \beta\in V\} \cup (U\cap\Sigma^\omega).$$

Note that UV containes all ω-sequences of U. The $*$-closure U^* of a subset U of Σ^∞ is defined by

$$U^* = U^0 \cup U^1 \cup U^2 \cup \ldots$$

where $U^0=\{\lambda\}$, and $U^i=U^{i-1}U$ for $i>0$. That is, U^* is the set consisting of the empty sequence λ and all sequences formed by concatenating a finite number of sequences in U. We define the ω-closure U^ω of U by

$$U^\omega = \underbrace{(U-\{\lambda\})(U-\{\lambda\})(U-\{\lambda\})\ldots}_{\text{countably infinite times}}$$

U^ω is the set of all ω-sequences obtained by concatenating a finite or countably infinite number of sequences in U. We let $U^\infty=U^*\cup U^\omega$.

2.1 Extended Regular Expressions

[Definition 1] The class of extended regular expressions (EREs) over Σ is defined recursively as follows:

1. λ, ϕ and Σ are EREs over Σ.
2. Every symbol $e\in\Sigma$ is an ERE over Σ.
3. If R_1 and R_2 are EREs over Σ, so are $(R_1\cup R_2)$, $(R_1\cap R_2)$, (R_1-R_2), $(R_1\cdot R_2)$, $(R_1)^*$, $(R_1)^\omega$, $(R_1)^\infty$, $(\overset{*}{\sim}R_1)$, $(\overset{\omega}{\sim}R_1)$, $(\overset{\infty}{\sim}R_1)$ and $(\mathrm{pre}(R_1))$.
4. For an ERE R_1 and $L\subseteq\Sigma$, $\Pi_L(R_1)$ is an ERE. □

Every ERE R over Σ describes a subset $\langle R\rangle$ of Σ^∞, as defined below.

[Definition 2]

1. $\langle\lambda\rangle=\{\lambda\}$, $\langle\phi\rangle=\phi$ and $\langle\Sigma\rangle=\Sigma$.
2. $\langle e\rangle=\{e\}$ where $e\in\Sigma$.
3. Let R_1 and R_2 be EREs.

   $\langle(R_1\cup R_2)\rangle=\langle R_1\rangle\cup\langle R_2\rangle$

   $\langle(R_1\cap R_2)\rangle=\langle R_1\rangle\cap\langle R_2\rangle$

   $\langle(R_1-R_2)\rangle=\langle R_1\rangle-\langle R_2\rangle$ ("$-$" denotes set difference.)

   $\langle(R_1\cdot R_2)\rangle=\langle R_1\rangle\langle R_2\rangle$

   $\langle(R_1)^*\rangle=\langle R_1\rangle^*$

$<(R_1)^{\omega}>=<R_1>^{\omega}$

$<(R_1)^{\infty}>=<R_1>^{\infty}$

$<(\overset{*}{\sim}R_1)>=\Sigma^{*}-<R_1>$

$<(\overset{\omega}{\sim}R_1)>=\Sigma^{\omega}-<R_1>$

$<(\overset{\infty}{\sim}R_1)>=\Sigma^{\infty}-<R_1>$

$<(\mathrm{pre}(R_1))>=\{\alpha|\alpha$ is a prefix of some $\beta\in<R_1>.\}$

4. For an ERE R_1 and $L\in\Sigma$,

$<\Pi_L(R_1)>=\{\pi_L(\alpha)|\alpha\in<R>\}$ ($\pi_L(\alpha)$ is the sequence obtained from α by deleting all occurrences of symbols of Σ-L.) □

Parentheses may be omitted from an ERE according to convention if their omittion causes no confusion. By extending the result of McNaughton [6], we obtain the following proposition (the proof is omitted).

[Proposition 1] There is an algorithm to determine whether $<R> \subseteq <R'>$ holds for arbitrary EREs R and R′. □

2.2 Abbreviations for EREs

We now introduce some useful notations to be used as abbreviations for EREs (see Table 1). Table 1 is not meant to be an exclusive list of such abbreviations, and whenever necessary we will define other notations that will make EREs more concise and readable.

Table 1 Examples of Abbreviations for EREs over Σ

(e_1 and e_2 are symbols of Σ. R_1 and R_2 are EREs, possibly in abbreviated forms.)

ERE (possibly in an abbreviated form)	Abbreviation
$\Sigma^{*}\cdot e_1\cdot\Sigma^{\infty}$	eventually e_1
$\Sigma^{*}\cdot e_1$	endwith e_1
$e_1\cdot\Sigma^{\infty}$	startwith e_1
$(\Sigma-e_1)^{\infty}\cup(\Sigma-e_1)^{*}\cdot e_2\cdot\Sigma^{*}\cdot e_1\cdot\Sigma^{\infty}$	e_1 onlyafter e_2
(eventually e_1)∪(eventually e_2)	e_1 unless e_2
$\overset{\infty}{\sim}((R_1\cap\Sigma^{*})\cdot(\overset{\infty}{\sim}R_2))$	$R_1 \to\!\to R_2$
$(R_1\cap\Sigma^{*})\cdot\Sigma^{\infty}$	$\Diamond_{\mathrm{prefix}}\ R_1$ (pronounced "some prefix belongs to $<R_1>$")
$\Sigma^{*}\cdot R_1$	$\Diamond_{\mathrm{postfix}}\ R_1$ ("some postfix belongs to $<R_1>$")
$\overset{\infty}{\sim}((\overset{*}{\sim}R_1)\cdot\Sigma^{\infty})$	$\Box_{\mathrm{prefix}}\ R_1$ ("every prefix belongs to $<R_1>$")
$\overset{\infty}{\sim}(\Sigma^{*}\cdot(\overset{\infty}{\sim}R_1))$	$\Box_{\mathrm{postfix}}\ R_1$ ("every postfix belongs to $<R_1>$")

Note that some of the notations are similar to those found in temporal logic [5]. They can be used to make specifications containing temporal assertions simpler and easier to understand. For example, $<R_1 \to\!\to R_2>$ consists of all sequences α such that for every prefix β of α, $\beta\in<R_1>$ implies $\beta\backslash\alpha\in<R_2>$. ERE

$$(\text{endwith } e_1) \twoheadrightarrow ((\text{eventually } e_2) \cap (e_1 \text{ onlyafter } e_2))$$

characterizes all sequences in which whenever e_1 occurs, then after that (1) e_2 must eventually occur, and (2) e_1 can occur next only after an occurrence of e_2. $\langle \Box_{\text{prefix}} R_1 \rangle$ is the set of all sequences α such that every prefix of α belongs to $\langle R_1 \rangle$.

3. VERIFICATION OF PROPERTIES

Let Σ be a finite nonempty set of events that may take place in a system whose properties we are concerned with. An event sequence α of the system is said to be *incomplete* if (1) α is finite and (2) α is a prefix of some other event sequence of the system. Suppose that EREs R and R' over Σ are given such that:

(1) $\langle R \rangle$ is the set of all non-incomplete event sequences of the system.

(2) $\langle R' \rangle$ is a set of event sequences that have some "desirable" properties, such as liveness and safety.

Then the system satisfies the properties specified by R' if and only if

$$\langle R \rangle \subseteq \langle R' \rangle \qquad (*)$$

holds. That is, the problem of deciding whether the system has certain properties can be reduced to the containement problem for two EREs, which is decidable by Proposition 1. Therefore, at least in principle, it is possible to construct an automatic verifier for testing (*). At the same time, since (*) trivially holds if $\langle R \rangle = \phi$ (say, when the specification of the system given as ERE R "contains a contradiction"), in addition to the test for (*), we also have to show the containement of the form

$$\langle R'' \rangle \subseteq \langle R \rangle \qquad (**)$$

where R'' is an ERE over Σ such that, for example, $\langle R'' \rangle$ is a set of non-incomplete event sequences that must necessarily be allowed in the system.

4. EXAMPLE

We illustrate the use of EREs with an example of decentralized daisy chain arbiters.

4.1 Decentralized Daisy Chain Arbiter

A decentralized daisy chain arbiter (or simply an arbiter) [7,8] we consider here is a digital circuit consisting of a number of identical cells C_i, $0 \le i \le K-1$ and $K \ge 2$, connected by token transfer wires x_i in a ring structure, as shown in Fig. 1. Wire x_i is an output of C_i and an input to C_{i+1} (operations on subscripts i should be taken modulo K). Associated with each cell C_i is a process P_i which occasionally requests the exclusive use of a common resource or device such as a bus that can not be used by more than one process simultaneously. Process P_i and cell C_i communicate by request wire r_i and acknowledge wire a_i. Wire r_i (or a_i) is an output of P_i (or C_i), and an input to C_i (or P_i).

P_i sets r_i to logical value HIGH when a request for the exclusive use of the common resource occurs. C_i can acknowledge the request of P_i by setting a_i to HIGH provided that the "token" (a privilege) resides in C_i. There is exactly one token in the system, and the cells circulate the token in a fixed direction by supplying a short pulse to wires x_i; C_i transfers the token to C_{i+1} by supplying a short pulse to wire x_i. Initially the token resides in C_0. Process P_i and cell C_i communicate in so-called the handshake manner in which the following conditions are satisfied:

(HS0) Initially, both r_i and a_i are LOW.

(HS1) P_i sets r_i to HIGH (or LOW) only if a_i is LOW (or HIGH).

(HS2) C_i sets a_i to HIGH (or LOW) only if r_i is HIGH (or LOW).

P_i can start using the resource when a_i becomes HIGH, and when P_i releases the resource, r_i is set to LOW. By (HS1), once r_i is changed to LOW, P_i can set it to HIGH again only after C_i sets a_i to LOW. Therefore the resource is used by P_i only while both r_i and a_i are HIGH.

Any arbiter must have the following two properties.

(No Deadlock/Lockout) Every request is acknowledged eventually, and whenever process P_i releases the resource, C_i eventually becomes ready to receive the next request of P_i. That is, whenever r_i changes to HIGH (or LOW), a_i eventually becomes HIGH (or LOW).

(Mutual Exclusion) No two processes use the common resource simultaneously. That is, at any moment, there is at most one i, $0 \le i \le K-1$, such that r_i and a_i are both HIGH.

4.2 Specification

We present an ERE *D* describing the behavior of the arbiter. The set of events is given by

$$\Sigma = \{\hat{x}_0, \dots, \hat{x}_{K-1}, r_0\uparrow, \dots, r_{K-1}\uparrow, r_0\downarrow, \dots r_{K-1}\downarrow, a_0\uparrow, \dots, a_{K-1}\uparrow, a_0\downarrow, \dots, a_{K-1}\downarrow\}$$

where each event represents the following:

$\hat{x}_i$: C_i transfers the token to C_{i+1} by supplying a pulse to x_i.

$r_i\uparrow$ (or $r_i\downarrow$): P_i sets the value of request wire r_i to HIGH (or LOW).

$a_i\uparrow$ (or $a_i\downarrow$): C_i sets the value of acknowledge wire a_i to HIGH (or LOW).

We assume that no two events take place simultaneously.

First, we introduce several useful notations to be used as abbreviations for EREs: For each wire $v \in \{r_0, r_1, \dots, r_{K-1}, a_0, a_1, \dots, a_{K-1}\}$,

$$\mathrm{low}(v) \equiv (\Sigma - v\uparrow)^{\infty} \cup \Sigma^{*} \cdot v\downarrow \cdot (\Sigma - v\uparrow)^{\infty} \text{ and}$$

$$\mathrm{high}(v) \equiv \Sigma^{*} \cdot v\uparrow \cdot (\Sigma - v\downarrow)^{\infty},$$

where $v\uparrow$ and $v\downarrow$ represent $r_i\uparrow$ and $r_i\downarrow$ (or $a_i\uparrow$ and $a_i\downarrow$), respectively, if $v = r_i$ (or a_i), for each i. From the interpretation of the events $r_i\uparrow$, $r_i\downarrow$ and the assumption that the value of wire r_i is LOW initially, it should be clear that, when an event sequence α has occured, wire r_i has the value LOW (or HIGH) if and only if $\alpha \in \langle \mathrm{low}(r_i) \rangle$, (or $\alpha \in \langle \mathrm{high}(r_i) \rangle$). A similar observation applies to wires a_i. Further, we define:

$$\mathrm{priv}_0 \equiv (\Sigma - \hat{x}_0)^{\infty} \cup \Sigma^{*} \cdot \hat{x}_{K-1} \cdot (\Sigma - \hat{x}_0)^{\infty} \text{ and}$$

$$\mathrm{priv}_i \equiv \Sigma^{*} \cdot \hat{x}_{i-1} \cdot (\Sigma - \hat{x}_i)^{\infty}, \quad 1 \le i \le K-1.$$

It should be clear that, when an event sequence α has occurred, cell C_i has the token if and only if $\alpha \in \langle \mathrm{priv}_i \rangle$.

ERE *D* over Σ describing the set of all possible non-incomplete event sequences of the arbiter is given by

$$D = D_0 \cap D_1 \cap \ldots \cap D_{K-1} \cap W$$

where D_i, $0 \le i \le K-1$, is an ERE over Σ describing the possible event sequences with respect to the behavior of cell C_i. The interpretations of events $r_i\uparrow$, $r_i\downarrow$, $a_i\uparrow$ and $a_i\downarrow$ are reflected in ERE

$$W = \mathrm{WIRE}(r_0) \cap \ldots \cap \mathrm{WIRE}(r_{K-1}) \cap \mathrm{WIRE}(a_0) \cap \ldots \cap \mathrm{WIRE}(a_{K-1})$$

where for each $v \in \{r_0, \ldots, r_{K-1}, a_0, \ldots, a_{K-1}\}$, WIRE(v) is the following ERE ($v\uparrow$ is $r_i\uparrow$ if $v = r_i$, etc.):

$$\begin{aligned} \mathrm{WIRE}(v)\colon & \Box_{\mathrm{prefix}}(\overset{*}{\sim}(\mathrm{low}(v)\cdot v\downarrow)) \\ & \cap\ \Box_{\mathrm{prefix}}(\overset{*}{\sim}(\mathrm{high}(v)\cdot v\uparrow)). \end{aligned}$$

For example, $\mathrm{WIRE}(a_i)$ says that $a_i\uparrow$ (or $a_i\downarrow$) occurs only if the value of wire a_i is LOW (or HIGH).

The description D_i of cell C_i, $0 \le i \le K-1$, is given by

$$D_i = S_i \cup ((\mathrm{pre}(S_i)\cdot(r_i\uparrow \cup r_i\downarrow \cup \hat{x}_{i-1})) \cap \overset{*}{\sim}\mathrm{valid}_i)\cdot\Sigma^\infty$$

where ERE S_i and abbreviation valid_i are shown below:

$$S_i = \mathrm{valid}_i \cap E_i$$

$$\begin{aligned} \mathrm{valid}_i \equiv\ & \Box_{\mathrm{prefix}}(\overset{*}{\sim}(\mathrm{priv}_i\cdot\hat{x}_{i-1})) \\ & \cap\ \Box_{\mathrm{prefix}}(\overset{*}{\sim}(\mathrm{high}(a_i)\cdot r_i\uparrow)) \\ & \cap\ \Box_{\mathrm{prefix}}(\overset{*}{\sim}(\mathrm{low}(a_i)\cdot r_i\downarrow)) \end{aligned}$$

$$E_i = e_i^1 \cap e_i^2 \cap \ldots \cap e_i^9.$$

EREs e_i^1 through e_i^9 are:

e_i^1: $(\mathrm{priv}_i \cap (\mathrm{low}(a_i) \cup \mathrm{low}(r_i))) \twoheadrightarrow (\hat{x}_i \text{ unless } a_i\uparrow)$

e_i^2: $\Box_{\mathrm{prefix}}(\overset{*}{\sim}((\overset{*}{\sim}\mathrm{priv}_i)\cdot\hat{x}_i))$

e_i^3: $\Box_{\mathrm{prefix}}(\overset{*}{\sim}((\mathrm{high}(a_i) \cap \mathrm{high}(r_i))\cdot\hat{x}_i))$

e_i^4: $\Box_{\mathrm{prefix}}(\overset{*}{\sim}((\overset{*}{\sim}\mathrm{priv}_i)\cdot a_i\uparrow))$

e_i^5: $\Box_{\mathrm{prefix}}(\overset{*}{\sim}(\mathrm{low}(r_i)\cdot a_i\uparrow))$

e_i^6: $\Box_{\mathrm{prefix}}(\overset{*}{\sim}(\mathrm{high}(r_i)\cdot a_i\downarrow))$

e_i^7: endwith $r_i\downarrow \twoheadrightarrow$ eventually $a_i\downarrow$

e_i^8: $\overset{\infty}{\sim}(\Sigma^*\cdot((\Sigma-\hat{x}_i)^*\cdot a_i\uparrow)^\omega)$

e_i^9: $\overset{\infty}{\sim}(\Sigma^*\cdot r_i\uparrow\cdot((\Sigma-a_i\uparrow)^*\cdot\hat{x}_{i-1})^\omega)$.

Intuitively, the behavior of cell C_i is specified in E_i; the intended meaning of E_i is as follows.

- e_i^1: If cell C_i has the token and process P_i is not using the resource, then C_i must eventually either acknowledge a request of process P_i, or transfer the token to C_{i+1}.
- e_i^2: C_i can transfer the token to C_{i+1} only if the token resides in C_i.
- e_i^3: C_i can transfer the token to C_{i+1} only if P_i is not using the resource.
- e_i^4: C_i can ackowledge a request of P_i only if it has the token.
- e_i^5,e_i^6: C_i communicates with P_i in the handshake manner.
- e_i^7: If P_i "releases" the resource, C_i will eventually be ready to accept the next request of P_i.
- e_i^8: C_i is not allowed to ackowledge the requests of P_i infinitely often, without transferring the token to C_{i+1}.
- e_i^9: C_i is not allowed to "neglect" a request of P_i, while receiving the token infinitely many times.

E_i itself, however, cannot be adopted as a specification of cell C_i, since in practice, cell circuits are designed to work properly (say, as specified in E_i) only if its "neighbors" (such as C_{i-1} and P_i in our example) also "behave properly". In this particular example, abbreviation valid_i reflects the idea of "proper behavior" of the neighbors of C_i: valid_i characterizes event sequences in which (1) C_{i-1} does not transfer the token to C_i if C_i already has one, and (2) P_i respects the handshake rule for communicating with C_i.

Then, ERE $S_i = \mathrm{valid}_i \cap E_i$ specifies how C_i must work if the behavior of its neighbors is proper. The rest of D_i says that C_i must obey E_i as long as valid_i is satisfied (i.e., C_i must work properly as long as its neighbors also do), but once valid_i is violated by an inproper occurrence of $r_i\uparrow$, $r_i\downarrow$ or $\hat{x}_{i-1}$, then the behavior of C_i thereafter is totally arbitrary.

4.3 Verification

We prove that the arbiter described by D works correctly if (1) the processes P_i obey the handshake rule, and (2) no process continues to use the resource without releasing it forever. This condition on the behavior of the processes P_i is described in ERE

$$F = F_0 \cap F_1 \cap \ldots \cap F_{K-1}$$

where for each i, $0 \le i \le K-1$,

$$F_i = f_i^1 \cap f_i^2 \cap f_i^3$$

$$f_i^1\colon \Box_{\mathrm{prefix}}(\overset{*}{\sim}(\mathrm{high}(a_i)\cdot r_i\uparrow))$$

$$f_i^2\colon \Box_{\mathrm{prefix}}(\overset{*}{\sim}(\mathrm{low}(a_i)\cdot r_i\downarrow))$$

$$f_i^3\colon (\mathrm{endwith}\ a_i\uparrow \cap \mathrm{high}(r_i)) \twoheadrightarrow (r_i\downarrow\ \mathrm{unless}\ a_i\downarrow).$$

f_i^1 and f_i^2 say that P_i obeys the handshake rule, and f_i^3 says that P_i eventually releases the resource (provided that C_i obeys the handshake rule).

Lemma 1 says that the token keeps on moving among the cells in the system consisting of the arbiter described by D and the processes given by F.

[Lemma 1] Let $X=\{\hat{x}_0,\hat{x}_1,\ldots,\hat{x}_{K-1}\}$. For any $\alpha \in \langle D \cap F\rangle$, $\pi_X(\alpha)=(\hat{x}_0\hat{x}_1\ldots\hat{x}_{K-1})^{\omega}$. □

Lemma 1 is used for proving Theorems 1 and 2 which show that the system consisting of the arbiter and the processes has two fundamental properties.

[Theorem 1] (No Deadlock/Lockout) For any i, $0 \le i \le K-1$,

(1) $\langle D \cap F\rangle \subseteq \langle \mathrm{endwith}\ r_i\uparrow \twoheadrightarrow \mathrm{eventually}\ a_i\uparrow\rangle$, and

(2) $\langle D \cap F\rangle \subseteq \langle \mathrm{endwith}\ r_i\downarrow \twoheadrightarrow \mathrm{eventually}\ a_i\downarrow\rangle$. □

[Theorem 2] (Mutual Exclusion) For any distinct i and j, $0 \le i,j \le K-1$,

$$\langle D \cap F\rangle \subseteq \langle \Box_{\mathrm{prefix}}(\overset{*}{\sim}(\mathrm{high}(a_i) \cap \mathrm{high}(r_i) \cap \mathrm{high}(a_j) \cap \mathrm{high}(r_j)))\rangle. \ \Box$$

Finally, we have to prove (at least) that $\langle D \cap F\rangle$ is not empty, since Theorems 1 and 2 only assert that there is no "undesirable" sequence in $\langle D \cap F\rangle$.

Consider another description D' of an arbiter, in which the set of possible non-incomplete event sequences is more restrcted than in D:

$$D' = D'_0 \cap D'_1 \cap \dots \cap D'_{K-1} \cap W$$

$$D'_i = S'_i \cup ((\mathrm{pre}(S'_i)\cdot(r_i\uparrow \cup r_i\downarrow \cup \hat{x}_{i-1})) \cap \overset{*}{\sim}\mathrm{valid}_i)\cdot R$$

$$S'_i = \mathrm{valid}_i \cap E'_i$$

$$E'_i = E_i \cap e_i^{10} \cap e_i^{11}$$

$$e_i^{10}\text{: } (\mathrm{high}(r_i)\cdot\hat{x}_{i-1}) \twoheadrightarrow (\hat{x}_i \text{ onlyafter } a_i\uparrow)$$

$$e_i^{11}\text{: } (\mathrm{priv}_i\cdot r_i\uparrow) \twoheadrightarrow (a_i\uparrow \text{ onlyafter } \hat{x}_i)$$

R is an ERE over Σ that describes what happens if the neighbors of C_i violate valid_i. e_i^{10} and e_i^{11} state that, when cell C_i has the token, C_i acknowledges a request of process P_i before transferring the token to C_{i+1}, if and only if the request has occurred before the token arrives. (Actually, e_i^8 and e_i^9 can be omitted in D'_i, since they are implied by e_i^{10} and e_i^{11}.) Clearly $\langle D' \cap F\rangle \subseteq \langle D \cap F\rangle$ holds for any R.

Theorem 3 states that the description $D' \cap F$ contains no contradiction; i.e., event sequences with desirable properties (of Theorems 1 and 2) actually occur in the arbiter specified by $D' \cap F$ (and thus by $D \cap F$).

[Theorem 3]

(1) $(\hat{x}_0\hat{x}_1\dots\hat{x}_{K-1})^{\omega} \in \langle D' \cap F\rangle$ (hence $\langle D' \cap F\rangle \neq \phi$).

(2) Let α be any prefix of a sequence in $\langle D' \cap F\rangle$ such that $\alpha \in \langle\mathrm{low}(r_i)\rangle \cap \langle\mathrm{low}(a_i)\rangle$ (note that the null prefix λ satisfies this). Then $\alpha\cdot r_i\uparrow$ is also a prefix of some sequence in $\langle D' \cap F\rangle$. □

The proofs of Lemma 1 and the theorems cannot be included here for lack of space. They have been proved not by an automatic decision method based upon Proposition 1, but by using induction on the length of event sequences and a number of simple propositions and lemmas such as:

(1) $\langle\mathrm{pre}(R_1 \cap R_2)\rangle \subseteq \langle\mathrm{pre}(R_1)\rangle$.

(2) $\langle\mathrm{pre}(\Box_{\mathrm{prefix}}(R))\rangle \subseteq \langle R\rangle$.

(3) For $e_1, e_2 \in \Sigma$, if $\alpha \in \langle R\rangle$ and $\alpha e_1 \in \langle \overset{*}{\sim}(R\cdot e_2)\rangle$, then $e_1 \neq e_2$.

(4) For $\alpha \in \langle\Sigma^*\rangle$, $\alpha \notin \langle R\rangle$ if and only if $\alpha \in \langle \overset{*}{\sim}R\rangle$.

(5) For each $v \in \{r_0, \dots, r_{K-1}, a_0, \dots, a_{K-1}\}$, $\alpha \in \langle\mathrm{low}(v)\rangle$ if and only if $\alpha \notin \langle\mathrm{high}(v)\rangle$.

(6) For each $v \in \{r_0, \dots, r_{K-1}, a_0, \dots, a_{K-1}\}$, if $\alpha \in \langle\mathrm{high}(v)\rangle$, then $\alpha = \alpha_1 v\uparrow\alpha_2$ for some $\alpha_1 \in \langle\Sigma^*\rangle$ and $\alpha_2 \in \langle(\Sigma - v\uparrow - v\downarrow)^*\rangle$.

(7) For any $\alpha \in \langle\mathrm{pre}(D)\rangle$, either $\alpha \in \langle\mathrm{pre}(E_i)\rangle$ or $\alpha \in \langle \overset{*}{\sim}\mathrm{valid}_i\rangle$, $0 \le i \le K-1$.

This is because the number of cells, K, is left unspecified in $D \cap F$ and $D' \cap F$. Mechanical proofs are possible for any fixed value of K (at least in principle), and in fact, for the case of $K=3$, we have proved Theorems 1 and 2 by using an algorithm for testing the containment "$\subseteq$" of two EREs. As K becomes large, however, mechanical proofs tend to be computationally infeasible.

4.4 Application

Fig. 2 shows a circuit to be used as a cell in an arbiter. Before the arbiter begins to operate, RESET_i is set to 0 (=LOW) for a while at all cells simultaneously in order to clear all flip-flops. After this, the value of RESET_i is always 1 (=HIGH). The value of START_i is 1, except when a negative pulse ($\urcorner\!\sqcup\!\ulcorner$) is supplied to START_i at CELL_0 in order to start the system. The transfer of the token is accomplished by supplying a negative pulse to wire x_i.

Assume that, in the circuit, the probability of malfunctions is negligibly small even when two "events" take place simultaneously. Then the behavior of the arbiter consisting of the circuit can be studied in terms of event sequences. Consider the arbiter consisting of the circuit, and let ARBITER be the set of non-incomplete event sequences that occur when processes P_i behave properly as specified in F. It should not be difficult to see that for some R we have ARBITER = $\langle D' \cap F \rangle$. So by Theorems 1-3 we conclude that the circuit constructed from the circuit of Fig. 2 works correctly as an arbiter.

5. CONCLUSION

As is seen from the discussions and the example in the previous sections, using EREs for specifying systems and verifying properties of systems has the following advantages:

(1) The specification of a system is easy to understand, since usually it can be obtained as a combination of simple EREs. Each one of these simple EREs represents a "local" restriction on the occurrences of particular events.

(2) The specification of a system can be given independently of the inner structure of the system, and the verification of properties can be carried out without knowing how the system is implemented.

(3) The problem of verifying properties of a system can be formulated as the containement problem for two EREs. Since the latter is solvable, it is decidable whether a system satisfies given propperties.

A specification and verification of a two-process arbiter in terms of a class of temporal logic are found in Bochmann [1]. The decidability of the temporal logic is not discussed in [1], while we have proved that the verification problem is decidable in EREs. The specification and the verification given in [1] concern a particular logic circuit, while we have proved that any logic circuit satisfying the requirements for a cell can be used for constructing an arbiter.

REFERENCES

[1] G.V. Bochmann, "Hardware Specification with Temporal Logic: An Example," Transactions on Computers C-31 (1982), 223-231.
[2] Y. Choueka, "Theories of Automata on ω-Tapes: A Simplified Approach," Journal of Computer and System Sciences 8 (1974), 117-141.
[3] B.T. Heilpern and S.S. Owicki, "Modular Verification of Computer Communication Protocols," Transactions on Communications COM-31 (1983), 56-68.
[4] L. Landweber, "Decision Problems for ω-Automata," Mathematical Systems Theory 3 (1969), 376-384.
[5] Z. Manna and A. Pnueli, "Verification of Concurrent Programs: The Temporal Framework," in The Correctness Problem in Computer Science, Academic Press (1981).
[6] R. McNaughton, "Testing and Generating Infinite Sequences by a Finite Automaton," Information and Control 9 (1966), 521-530.
[7] W.W. Plummer, "Asynchronous Arbiters," Transactions on Computers C-21 (1972), 37-42.
[8] K.J. Thurber et al., "A Systematic Approach to the Design of Digital Bussing

Structures," in AFIPS Conference Proceedings (1972), 719-740.

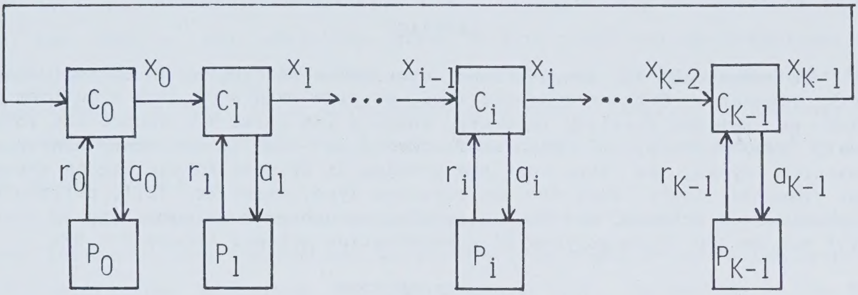
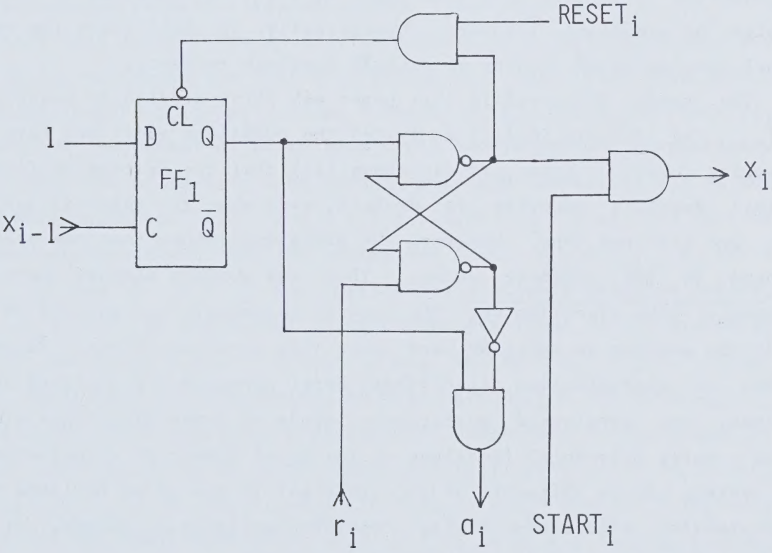

C0
x0
C1
x1
...
xi-1
Ci
xi
...
xK-2
CK-1
xK-1
r0
a0
r1
a1
ri
ai
rK-1
aK-1
P0
P1
Pi
PK-1

Fig. 1 Decentralized Daisy Chain Arbiter


RESETi
CL
1
D
Q
FF1
xi-1
C
Q̄
xi
ri
ai
STARTi

FF$_1$: Edge-Triggered D Flip-Flop

Fig. 2 A Circuit of Cell C_i

COMPLEXITY OF OPTIMAL DEADLOCK RECOVERY

JOSEPH Y-T. LEUNG
Department of Electrical Engineering and Computer Science
Northwestern University
Evanston,Illinois 60201

ABSTRACT

We consider the computational complexity of finding an optimal deadlock recovery. For one resource type, we give an O(n*C) time algorithm to find an optimal deadlock recovery, where n and C are the number and total cost, respectively, of the deadlocked jobs. For an arbitrary number of resource types, we show that the problem is NP-hard in the strong sense. Our results imply that for one resource type, there are fully polynomial approximation schemes, and for an arbitrary number of resource types, there will not be any fully polynomial approximation schemes unless P = NP.

1. INTRODUCTION

The problem of system deadlock has been studied for a long time by researchers in the operating systems area [2-4,7-9,11,13,15,16]. Recently, the problem has received renewed interest from researchers in the database systems area [1,6,12]. Previous researchers have approached this problem from three different angles: prevention [2,8,15,16], avoidance [2,7,13,15] and recovery [11,15,16]. In this paper we shall be concerned only with the problem of deadlock recovery. Specifically, we shall study the computational complexity of finding an optimal deadlock recovery.

The topic discussed in this paper was first studied by Leung and Lai [11]. The results in [11] motivated the questions addressed here in the following ways: First, it was shown [11] that the problem of finding an optimal deadlock recovery is NP-hard, even when the physical system has only one resource type. However, the proof only shows that the problem is NP-hard in the ordinary sense. Thus, one wonders whether there is any polynomial-time algorithm when the numeric parameters are encoded in unary, or if the problem is still NP-hard under this encoding scheme. Secondly, a number of approximation algorithms were proposed and analyzed in [11]. However, the worst-case performance bounds of these algorithms are quite large - being polynomial functions of the total number of resource units in the system. It is therefore of some interest to see if we can come up with approximation algorithms having constant worst-case bounds, or better still, one which gives ε - approximate solutions.

The main results of this paper are: 1) For systems having one resource type, there is an O(n*C) time algorithm to find an optimal deadlock recovery, where n and C are the number and total cost, respectively, of the deadlocked jobs. 2) For arbitrary number of resource types, the problem of finding an optimal deadlock recovery is NP-hard in the strong sense.

There are several implications we can draw upon our results. First, for one resource type, there are pseudo-polynomial time algorithms to find an optimal deadlock recovery, and for arbitrary number of resource types, no such algorithms exist unless P = NP. Secondly, for one resource type, we can combine the algorithm given in this paper and the techniques of rounding and scaling discussed in [5, 10] to devise polynomial-time algorithms yielding ε - approximate solutions. Finally, if the cost of each job is the same, then the running time of the algorithm alluded in (1) is reduced to $O(n^2)$.

In the next section we shall give a formal definition of the problem and introduce some notations. In section 3 we shall describe the pseudo-polynomial time algorithm for one resource type. In section 4, the NP-hardness proof will be given. Finally, we shall draw some concluding remarks and suggest some open problems for future research in the last section.

2. THE MODEL

We shall use more-or-less the same notations as in [11]. Let the physical system contain k resource types $R_1,\ldots,R_k$ with $w_1,\ldots,w_k$ units of the respective types. Let $J_1,\ldots,J_n$ be a set of n deadlocked jobs and let c_i $(1 \leq i \leq n)$ denote the cost of aborting J_i. Associated with each J_i is a pair of vectors $p_i = (p_{i1},\ldots,p_{ik})$ and $q_i = (q_{i1},\ldots,q_{ik})$, where p_{ij} and q_{ij} denote the number of units of R_j held and requested, respectively, by J_i at the time of deadlock. We assume that each deadlocked job is holding one or more resources (i.e. $\sum_{j=1}^{k} p_{ij} \geq 1$ for each $1 \leq i \leq n$), and that no job requires more resources than the system has (i.e., $p_{ij} + q_{ij} \leq w_j$ for all $1 \leq i \leq n$ and $1 \leq j \leq k$). Define the available resource vector, $v = (v_1,\ldots,v_k)$, to be the vector of resources available in the system at the time of deadlock; i.e., $v_j = w_j - \sum_{i=1}^{n} p_{ij}$ for each $1 \leq j \leq k$. Since each J_i is deadlocked in the system, we must have $q_i \nleq v$ for each $1 \leq i \leq n$.[1]

With the above notation, a deadlock is completely characterized by the 6-tuple, $D = (w, J, c, P, Q, v)$, where $w = (w_1,\ldots,w_k)$, $J = \{J_1,\ldots,J_n\}$, $c = \{c_1,\ldots,c_n\}$, $P = [p_1,\ldots,p_n]^T$, $Q = [q_1,\ldots,q_n]^T$, and $v = (v_1,\ldots,v_k)$, subject to the conditions that $\sum_{j=1}^{k} p_{ij} \geq 1$, $p_i + q_i \leq w$, and $q_i \nleq v$ for all

---

[1] If $x = (x_1,\ldots,x_k)$ and $y = (y_1,\ldots,y_k)$ are two vectors of integers, then $x \leq y$ implies that $x_i \leq y_i$ for each $1 \leq i \leq k$, and $x \nleq y$ implies that there is an i such that $x_i > y_i$.

$1 \leq i \leq n$. Since we are assuming no a priori information on the dynamic resource usage of the jobs, we shall assume that a deadlocked job, once granted all its requested resources, will proceed to completion without any further request.

Let $D = (w, J, c, P, Q, v)$ be a deadlock. A set $J' \subseteq J$ is called a deadlock-breaking set if after aborting all jobs in J' and releasing all resources held by those jobs, all the jobs in $J-J'$ can proceed to completion in some sequence. The cost of the deadlock-breaking set J', denoted by $c(J')$, is defined to be $c(J') = \sum_{J_i \varepsilon J'} c_i$. An optimal deadlock-breaking set is a deadlock-breaking set with the minimum cost. We wish to devise algorithms which will take as input an arbitrary deadlock and output an optimal deadlock-breaking set.

3. PSEUDO-POLYNOMIAL TIME ALGORITHM

The algorithm reported in this section is based on dynamic programming techniques. We shall introduce some additional notations that are applicable to this section. For one resource type, we shall use the convention that the vector names stand for the single component of the vectors; i.e. p_i denotes the number of resources held by J_i, etc. Moreover, we assume that the jobs have been indexed in non-decreasing order of their request demands; i.e., $q_1 \leq q_2 \leq \ldots \leq q_n$. Let $C = \sum_{i=1}^{n} c_i$ denote the total cost of the deadlocked jobs. For each $1 \leq i \leq n$, we let r_i denote the total number of resources held by the first i jobs; i.e., $r_i = \sum_{j=1}^{i} p_j$. With these conventions and notations in mind, we can now present our algorithm.

The heart of our algorithm is to construct a table T with n rows and C + 1 columns. Each entry $T(i,j)$ has one of three possible values - "Y", "N" and "M." If $T(i,j)$ is "Y", then it means that there is a deadlock-breaking set $J' \subseteq \{J_1,\ldots,J_i\}$ with cost j such that when all jobs in J' are aborted, the jobs in $\{J_1,\ldots,J_i\} - J'$ can proceed to completion in some sequence. If $T(i,j)$ is "N" then no such deadlock-breaking set exists. Finally, if $T(i,j)$ is "M", then there is a deadlock-breaking set provided that the available resource, v, is increased by a certain "deficit." This deficit is denoted by $d(i,j)$ and is associated with the (i,j)th entry of the table whenever $T(i,j)$ is "M". the cost of an optimal deadlock-breaking set will be ℓ, where ℓ is the smallest integer such that $T(n,\ell) =$ "Y."

We now consider how to construct the table T. The first row of T can easily be computed as follows:

$$T(1,j) = \begin{cases} \text{"Y"} & \text{if } j = c_1 \\ \text{"M"} & \text{if } j = 0;\ d(1,0) = q_1 - v \\ \text{"N"} & \text{otherwise.} \end{cases}$$

For each $2 \leq i \leq n$ and $0 \leq j \leq C$, we compute $T(i,j)$ as follows. Let $A = T(i-1,j-c_i)$ and $B = T(i-1,j)$. If $j < c_i$, we assume A = "N." We have the following cases:

Case 1: A = "Y." $T(i,j)$ = "Y."

Case 2: A = "M." If $d(i-1,j-c_i) \leq p_i$, then $T(i,j)$ = "Y." Otherwise, we have the following subcases to consider.

Case 2.1: B = "Y." If $q_i \leq r_{i-1} + v$, then $T(i,j)$ = "Y." Otherwise, $T(i,j)$ = "M" and $d(i,j) = \min\{d(i-1,j-c_i) - p_i,\ q_i - (r_{i-1} + v)\}$.

Case 2.2: B = "M." $T(i,j)$ = "M" and $d(i,j) = \min\{d(i-1,j-c_i) - p_i,\ \max\{d(i-1,j),\ q_i - (r_{i-1} + v)\}\}$.

Case 2.3: B = "N." $T(i,j)$ = "M" and $d(i,j) = d(i-1,j-c_i) - p_i$.

Case 3: A = "N." We have the following subcases to consider:

Case 3.1: B = "Y." If $q_i \leq r_{i-1} + v$, then $T(i,j)$ = "Y;" otherwise, $T(i,j)$ = "M" and $d(i,j) = q_i - (r_{i-1} + v)$.

Case 3.2: B = "M." $T(i,j)$ = "M" and $d(i,j) = \max\{d(i-1,j),\ q_i - (r_{i-1} + v)\}$.

Case 3.3: B = "N." $T(i,j)$ = "N." ●

The computation of the first row of T is easy to see and requires no explanation. For $2 \leq i \leq n$ and $0 \leq j \leq C$, the value of $T(i,j)$ is a function of the values of $T(i-1,j-c_i)$ and $T(i-1,j)$, which corresponds to the case where J_i is aborted and the case where J_i is not aborted, respectively. If $T(i-1,j-c_i)$ = "Y" (Case 1), then there is a deadlock-breaking set J' with cost $j-c_i$ which will resolve the deadlock among the first i-1 jobs. Clearly, then, by adding J_i to J', we have a deadlock-breaking set with cost j which will resolve the deadlock among the first i jobs.

If Case 2 occurs, then there is a deadlock-breaking set J' with cost $j-c_i$ which will resolve the deadlock among the first i-1 jobs, provided that the available resource is increased by $d(i-1,j-c_i)$. Now, by including J_i into J', we have a deadlock-breaking set with cost j and the available resource is increased by p_i. Thus, if $d(i-1,j-c_i) \leq p_i$, then $J' \cup \{J_i\}$ will resolve the deadlock among the first i jobs. Otherwise, it will resolve the deadlock provided that the available resource is increased by $d(i-1,j-c_i) - p_i$. For this case, we have to examine the value of $T(i-1,j)$.

If Case 2.1 occurs, then there is a deadlock-breaking set J" with cost j which will resolve the deadlock among the first i-1 jobs. The amount of resources released by these i-1 jobs is exactly r_{i-1}. Thus, if $q_i \leq r_{i-1} + v$, then J" will resolve the deadlock among the first i jobs and has a cost

j. Otherwise, it is a deadlock-breaking set provided that the available resource is increased by $q_i - (r_{i-1} + v)$. Consequently, we will choose either $J' \cup \{J_i\}$ or J'' depending on which one has a smaller deficit.

If Case 2.2 occurs, then there is a deadlock-breaking set J'' with cost j which will resolve the deadlock among the first $i-1$ jobs, provided that the available resource is increased by $d(i-1,j)$. Now, in order for J'' to resolve the deadlock among the first i jobs, the available resource must be increased by an amount equal to the larger of $d(i-1,j)$ and $q_i - (r_{i-1} + v)$. This is because J_i must be the last one to proceed among all jobs in $\{J_1, \ldots, J_i\} - J''$, since it has the largest request demand. The deficit for J_i is $q_i - (r_{i-1} + v)$ and the deficit for the first $i-1$ jobs is $d(i-1,j)$. Therefore, we need to increase the available resource by an amount equal to the larger of $q_i - (r_{i-1} + v)$ and $d(i-1,j)$ in order for J'' to resolve the deadlock. Finally, we choose either $J' \cup \{J_i\}$ or J'' depending on which one has a smaller deficit.

From the above discussions, it is easy to see the computations of Case 2.3 and all of Case 3.

It is not difficult to show that our algorithm can be made to run in $O(n*C)$ time and $O(n*C)$ space.[2]

We note that our algorithm can be modified to solve a related problem. Suppose there is a subset $J' = \{J_{i_1}, J_{i_2}, \ldots, J_{i_X}\}$ of deadlocked jobs which must not be aborted because they incur high costs. Suppose further that we can resolve the deadlock without aborting any jobs in J'. (This can be tested by aborting all jobs in $\{J_1, \ldots, J_n\} - J'$ and checking to see if the deadlock can be resolved.) We wish to find an optimal deadlock-breaking set under this constraint. This problem can be solved by modifying our algorithm as follows: For each $1 \leq \ell \leq X$, $T(i_\ell, j)$ is computed by examining only $T(i_\ell - 1, j)$ for all $0 \leq j \leq C$. This will ensure J_{i_ℓ} $(1 \leq \ell \leq X)$ will not be included in any deadlock-breaking set. This approach is better than the naive method of assigning very large costs to jobs in J' and then running our algorithm. Furthermore, the total cost, C, is reduced to include only those jobs in $\{J_1, \ldots, J_n\} - J'$.

4. A REDUCTION

In this section we shall show that the problem of finding an optimal deadlock-breaking set is NP-hard in the strong sense. We show this by proving that the decision problem, defined below, is NP-complete in the strong sense.

---

[2] The optimal deadlock-breaking set can be obtained by storing pointers into the Table T.

DEADLOCK RECOVERY: Given a deadlock $D = (w, J, c, P, Q, v)$ and a positive integer b, is there a deadlock-breaking set $J' \subsetneq J$ such that $c(J') \leq b$?

Theorem 1: The DEADLOCK RECOVERY problem is NP-complete in the strong sense

Proof: The DEADLOCK RECOVERY problem is clearly in NP. To complete the proof, we shall reduce the 3-PARTITION problem to it. The 3-PARTITION problem has been shown to be NP-complete in the strong sense [5] and is defined as follows: Given a positive integer B and a set of 3m positive integers $A = \{a_1,\ldots,a_{3m}\}^3$ such that $\sum_{j=1}^{3m} a_j = mB$ and $B/4 < a_j < B/2$ for each $1 \leq j \leq 3m$, is there a partition of A into m disjoint sets $A_1,A_2,\ldots,A_m$ such that $\sum_{a_j \in A_i} a_j = B$ for each $1 \leq i \leq m$? (Note that the constraints on the a_j imply that each such A_i must contain exactly three elements from A.)

Given an instance $A = \{a_1,\ldots,a_{3m}\}$ and B of the 3-PARTITION problem, we construct an instance $D = (w, J, c, P, Q, v)$ and b of the DEADLOCK RECOVERY problem as follows: There will be 4m resource types $R_1,\ldots,R_{4m}$. The first m resource types have 2mB units each and the last 3m resource types have m units each; i.e., $w_j = 2mB$ for $1 \leq j \leq m$ and $w_j = m$ for $m + 1 \leq j \leq 4m$. The available resource vector, v, is the zero vector. There will be $3m^2 + 1$ jobs. For each a_i $(1 \leq i \leq 3m)$, we create m jobs J_i^ℓ $(1 \leq \ell \leq m)$. The last job is denoted by J_s. The costs as well as the resources held and requested by these $3m^2 + 1$ jobs are given in Figure 1. Finally, we choose $b = 3m$. It is not difficult to verify that the construction can be carried out in polynomial time.

We now show that the instance of the 3-PARTITION problem has a solution if and only if the instance of the DEADLOCK RECOVERY problem has a solution. Suppose that $A_i = \{a_{i_1},a_{i_2},a_{i_3}\}$ $(1 \leq i \leq m)$ is a solution to the 3-PARTITION problem. We shall show that there is a deadlock-breaking set J' with cost 3m. For each $A_i = \{a_{i_1},a_{i_2},a_{i_3}\}$ $(1 \leq i \leq m)$, we include $J_{i_1}^i, J_{i_2}^i$, and $J_{i_3}^i$ into J'. Since the cost of each of these jobs is 1, we have $c(J') = 3m$. Furthermore, the jobs in J' will release altogether B units of each of the first m types and 1 unit of each of the last 3m types. Thus, the available resource vector, v, becomes $[B,B,\ldots,B,1,1,\ldots,1]$.

---

[3] A is actually a multiset since we allow duplicate integers to be in A. For convenience in presentation, we shall refer to a multiset simply as a set.

Now, J_s can proceed and upon completion, it will release mB additional units of each of the first m types. Thus, the available resource vector becomes $[(m+1)B,(m+1)B,\ldots,(m+1)B,1,1,\ldots,1]$. From here on, the jobs in $\{J_1^1,\ldots,J_1^m,\ldots,J_{3m}^1,\ldots,J_{3m}^m\} - J'$ can proceed in any sequence. Therefore, J' is a deadlock-breaking set.

Conversely, suppose there is a deadlock-breaking set J' such that $c(J') \leq 3m$. We shall show that there is a solution to the 3-PARTITION problem. First, observe that J_s cannot be in J', since it has a cost of $3m + 1$. Thus, J' must consist of jobs from $\{J_1^1,\ldots,J_1^m,\ldots,J_{3m}^1,\ldots,J_{3m}^m\}$. Secondly, observe that in order for J' to be a deadlock-breaking set, it must release enough resources for J_s to proceed first. This is because all jobs in $\{J_1^1,\ldots,J_1^m,\ldots,J_{3m}^1,\ldots,J_{3m}^m\}$ require a lot of resources of the first m types, most of which are held by J_s. Thus, none of them can proceed until J_s has completed and released all the resources held by it. Consequently, the jobs in J' must altogether release at least B units of each of the first m types and 1 unit of each of the last 3m types (which are the resources requested by J_s). Thirdly, observe that for each $1 \leq i \leq 3m$, there is exactly one ℓ_i $(1 \leq \ell_i \leq m)$ such that $J_i^{\ell_i}$ (which is holding a_i units of R_{ℓ_i} and 1 unit of R_{m+i}) is in J'. This is because for each $1 \leq i \leq 3m$, each job J_i^{ℓ} $(1 \leq \ell \leq m)$ releases exactly 1 unit of R_{m+i}. Since J_s requires at least 1 unit of each of $R_{m+1},\ldots,R_{4m}$ and since $c(J') \leq 3m$, there is exactly one ℓ_i $(1 \leq \ell_i \leq m)$ such that $J_i^{\ell_i}$ is in J'. Finally, we observe that for each $1 \leq \ell \leq m$, there are exactly 3 jobs $J_{\ell_1}^{\ell}$, $J_{\ell_2}^{\ell}$, and $J_{\ell_3}^{\ell}$ in J' which release a total of B units of resource R_ℓ. Thus, $A_\ell = \{a_{\ell_1},a_{\ell_2},a_{\ell_3}\}$ $(1 \leq \ell \leq m)$ form a partition of A.

5. CONCLUDING REMARKS

In this paper we gave an $O(n*C)$ time algorithm to find an optimal deadlock recovery for systems having one resource type. The running time of the algorithm becomes $O(n^2)$ when the costs of the deadlocked jobs are the same. Moreover, our algorithm, when combined with the techniques of rounding and scaling, can be used as a basis for a fully polynomial approximation scheme [5]. For an arbitrary number of resource types, we showed that the problem is NP-hard in the strong sense. Thus, unless P = NP, there will not be any fully polynomial approximation schemes for the deadlock recovery problem.

We have not been able to determine the complexity of the deadlock recovery problem for fixed $k > 1$ resource types. For future research, we think it is worthwhile to settle this issue. It is tempting to try to extend our algorithm for one resource type to solve this problem. The "deficits" will now be maintained as k-tuples. The problem we encountered with this approach is how to update the "deficits" efficiently (in Cases 2.2 and 3.2). For one resource type the jobs can be ordered according to their request demands so that job J_i is the last job to proceed if the deadlock can be resolved among the first i jobs (without aborting J_i). When the number of resource types is larger than one, no such ordering appears possible. The nature of the problem suggests that an entirely different approach might be needed in order to devise a pseudo-polynomial time algorithm for this problem.

We indicated above that it is unlikely to have a fully polynomial approximation scheme for an arbitrary number of resource types. Our result, however, does not preclude the possibility of having a polynomial approximation scheme [5]. We have tried to extend the scheme proposed by Sahni [14] to develop a polynomial approximation scheme for the deadlock recovery problem. The idea is to examine all subsets of ℓ or fewer jobs. For each subset, we will successively add to it the job with the smallest cost until the deadlock can be resolved. Finally, we choose the deadlock-breaking set with the smallest cost. It can be shown, however, that the performance of this algorithm is not bounded by any fixed constant. It will be interesting to show whether or not there is any polynomial approximation scheme for this problem.

J	P	Q	c
J_1^1	$[a_1,0,\ldots,0,1,0,\ldots,0]$	$[(m+1)B,(m+1)B,\ldots,(m+1)B,1,1,\ldots,1]$	1
J_1^2	$[0,a_1,\ldots,0,1,0,\ldots,0]$	$[(m+1)B,(m+1)B,\ldots,(m+1)B,1,1,\ldots,1]$	1
	⋮	⋮	
J_1^m	$[0,0,\ldots,a_1,1,0,\ldots,0]$	$[(m+1)B,(m+1)B,\ldots,(m+1)B,1,1,\ldots,1]$	1
J_2^1	$[a_2,0,\ldots,0,0,1,\ldots,0]$	$[(m+1)B,(m+1)B,\ldots,(m+1)B,1,1,\ldots,1]$	1
J_2^2	$[0,a_2,\ldots,0,0,1,\ldots,0]$	$[(m+1)B,(m+1)B,\ldots,(m+1)B,1,1,\ldots,1]$	1
	⋮	⋮	
J_2^m	$[0,0,\ldots,a_2,0,1,\ldots,0]$	$[(m+1)B,(m+1)B,\ldots,(m+1)B,1,1,\ldots,1]$	1
	⋮	⋮	
J_{3m}^1	$[a_{3m},0,\ldots,0,0,0,\ldots,1]$	$[(m+1)B,(m+1)B,\ldots,(m+1)B,1,1,\ldots,1]$	1
J_{3m}^2	$[0,a_{3m},\ldots,0,0,0,\ldots,1]$	$[(m+1)B,(m+1)B,\ldots,(m+1)B,1,1,\ldots,1]$	1
	⋮	⋮	
J_{3m}^m	$[0,0,\ldots,a_{3m},0,0,\ldots,1]$	$[(m+1)B,(m+1)B,\ldots,(m+1)B,1,1,\ldots,1]$	1
J_s	$[mB,mB,\ldots,mB,0,0,\ldots,0]$	$[B,B,\ldots,B,1,1,\ldots,1]$	3m+1

Figure 1. Illustrating the Reduction in Theorem 1.

REFERENCES

1. W.W. Chu and G. Ohlamacher, "Avoiding deadlock in distributed data bases," *Proc. ACM Annual Conference*, San Diego, Cal., Nov., 1974, pp. 156-160.
2. E.G. Coffman, Jr. and P.J. Denning, *Operating Systems Theory*, Englewood Cliffs, New Jersey, Prentice-Hall, 1973, Chapter 2.
3. E.G. Coffman, Jr., M.J. Elphick and A. Shoshani, "System deadlocks," *Computing Surveys*, Vol. 3, No. 2, June 1971, pp. 67-78.
4. E.W. Dijkstra, "Cooperating sequential processes," in *Programming Languages*, F. Genuys, Ed., New York, Academic Press, 1968, pp. 43-112.
5. M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, San Francisco, Cal., Freeman and Co., 1979.
6. V.D. Gligor and S.H. Shattuck, "On deadlock detection in distributed systems," *IEEE Transactions on Software Engineering*, SE-6, Sept., 1980, pp. 435-440.
7. A.N. Habermann, "Prevention of system deadlock," *Commun. Assoc. Comput. Mach.*, Vol. 12, July 1969. pp. 373-377.
8. J.W. Havender, "Avoiding deadlock in multitasking systems," *IBM System Journal*, Vol. 7, 1968, pp. 74-84.
9. R.C. Holt, *On Deadlock in Computer Systems*, Ph.D. dissertation, Dept. of Computer Science, Cornell University, Ithaca, New York, Jan. 1971.
10. O.H. Ibarra and C.E. Kim, "Fast approximation algorithms for the knapsack and sum of subset problems," *J. Assoc. Comput. Mach.*, Vol. 22, 1975, pp. 463-468.
11. J. Y-T. Leung and E.K. Lai, "On minimum cost recovery from system deadlock," *IEEE Transactions on Computers*, Vol. C-28, Sept., 1979, pp. 671-677.
12. D.A. Menasce and R.R. Muntz, "Locking and deadlock detection in distributed data bases," *IEEE Transactions on Software Engineering*, SE-5, May 1979, pp. 195-202.
13. T. Minoura, "Deadlock avoidance revisited," *J. Assoc. Comput. Mach.*, Vol. 29, 1982, 1023-1048.
14. S. Sahni, "Approximate algorithms for the 0/1 knapsack problem," *J. Assoc. Comput. Mach.*, Vol. 22, 1975, pp. 115-124.
15. A.C. Shaw, *The Logical Design of Operating Systems*, Englewood Cliffs, New Jersey, Prentice-Hall, 1974, Chapter 8.
16. A. Shoshani, *Detection, Prevention and Recovery from Deadlocks in Multiprocess Multiple Resource Systems*, Ph.D. dissertation, Dept. of Electrical Engineering, Princeton University, Princeton, New Jersey, October 1969.

Preemptive Scheduling Of Uniform Processors With Memory*

Ten-Hwang Lai+ and Sartaj Sahni
University of Minnesota

Abstract

We consider the problem of preemptively scheduling n jobs on a system of m uniform processors to minimize the finish time.. Each job has a processing time and a memory requirement associated with it. Each processor has a speed and a memory capacity associated with it. Only the three cases m=2, m=3, and the processors fall into two classes are considered. For each of these cases, linear (in n) time algorithms are obtained.

Keywords and Phrases

preemptive scheduling, uniform processors, memories, complexity, finish time.

1. Introduction

Let $J = \{ J_i \mid 1 \le i \le n \}$ be a set of independent jobs. Associated with each job, J_i, is a 4-tuple (t_i, m_i, r_i, d_i) where t_i is the job's processing time $(t_i > 0)$, m_i is its memory requirement, r_i is its release time, and d_i is its due time. Let $P = \{ P_i \mid 1 \le i \le m \}$ be a set of m processors. With each processor, P_i, there is a 2-tuple (s_i, μ_i) associated. s_i is the *speed* of P_i and μ_i is its *memory capacity*. Job J_i (or simply job i) can be run on processor P_j (or simply processor j) iff $m_i \le \mu_j$. Furthermore, it takes P_j t_i/s_j time to completely run job J_i. P is said to be a *uniform processor system with memories*. If $s_1 = s_2 = \cdots = s_m$, then P is a system of *identical processors with memories*. If $s_1 = s_2 = \cdots = s_m$ and $\mu_1 = \mu_2 = \cdots = \mu_m$, P is simply a system of *identical processors*. Finally, if $\mu_1 = \mu_2 = \cdots = \mu_m$, P is called a system of *uniform processors*.

The $C_{\max}$ problem is that of determining a schedule S for which $C_{\max}(S)$ is minimum amongst all feasible schedules. Let S be a feasible schedule and let f_i be the finish time of job i in S. $L_i = f_i - d_i$ is the *lateness* of job i, $1 \le i \le n$. $L_{\max}(S) = \max_i \{L_i\}$ is the maximum lateness of any job in S. The $L_{\max}$ problem *is that of determining a schedule with minimum* $L_{\max}$.

When S is restricted to be a non-preemptive schedule, the $L_{\max}$ problem is known to be NP-hard even when m = 1 [3]. Under this same restriction, the $C_{\max}$ problem is NP-hard even when m = 2, $s_1 = s_2$, $\mu_1 = \mu_2$, and $r_1 = r_2 = \ldots = r_n$ [3]. In the remainder of this paper, we shall therefore be concerned only with schedules in which preemptions are permitted.

Suppose that P is a system of identical processors (i.e., $s_1 = s_2 = \ldots = s_m$ and $\mu_1 = \mu_2 = \ldots = \mu_m$). When all jobs have the same release time $(r_1 = r_2 = \ldots = r_n)$ a schedule that minimizes $C_{\max}$ can be obtained in O(n) time using McNaugton's algorithm [12]. When the release times are not necessarily the same, a schedule that minimizes $C_{\max}$ can be obtained in O(nm) time using the algorithm of Gonzalez and Johnson [5].

If P is a uniform processor system $(\mu_1 = \mu_2 = \ldots = \mu_m)$ and all jobs have the same release time, a schedule minimizing $C_{\max}$ may be obtained in O(n + mlogm) time using the algorithm of Gonzalez and Sahni [7]. When the release times are not necessarily the same, the algorithm of Sahni and Cho [14] may be used. This algorithm has a complexity of $O(m^2 n + mn\log n)$.

Scheduling on processor systems with memories has been studied by Kafura and Shen [8] as well as by Lai and Sahni [11]. Both these papers deal exclusively with a system of identical processors with memory. Kafura and Shen [8] develop an O(nlogm), $n \ge m$, algorithm for the $C_{\max}$ problem when all release times are the same. Lai and Sahni [11] consider the $L_{\max}$ problem when all release times are the same. Their algorithm is of complexity $O(k^2 n + n\log n)$ where k is the number of distinct due dates and $n \ge m$.

---

* This research was supported in part by the Office of Naval Research under contract N00014-80-C-0650.

+ Dr Lai is presently with the Computer Science Department at Ohio State University.

Both the general $C_{\max}$ and $L_{\max}$ problems can be formulated as linear programming problems as in [15]. The $C_{\max}$ problem for the case when all jobs have the same release time has been studied by Martel [16]. For any given c, he can determine in O(n logm) time whether there exists a feasible schedule with length c. However to find the minimum c for which a feasible schedule exists takes O(nL) time where L = $\sum_{1}^{n} \log t_i + \sum_{1}^{m} \log s_i$. L could be quite large and is always at least Ω(n+m).

In this paper, we consider scheduling systems of uniform processors with memories to minimize $C_{\max}$. Three special cases of $C_{\max}$ problem are considered in Section 2-4. Each of these cases assumes that all jobs are released at the same time (i.e., $r_1 = r_2 = \ldots = r_n$). In addition, the three cases, respectively, assume:

(1) m = 2

(2) m = 3

(3) The m processors can be partitioned into two classes A and B such that all processors in A have the same speed while all those in B have the same memory size. In addition, all processors in A have a larger memory size than those in B. So, for some ordering of the processors, it is the case that $\mu_1 \ge \mu_2 \ge \ldots \ge \mu_k > \mu_{k+1} = \ldots = \mu_m$ and $s_1 = s_2 = \ldots = s_k$.

For each of the above three cases, we develop linear time algorithms. These are faster than Martel's $\Omega(n^2)$ algorithms for these cases.

In order to establish the correctness of our algorithms, we shall make use of an important result from Sahni and Cho [13]. Let P_i, $1 \le i \le m$ be m processors with the same memory size. Let $s_i(t)$, $1 \le i \le m$ be m nondecreasing functions; $s_i(t)$ gives the speed at which P_i may operate at time t. $\{P_1, P_2, \ldots, P_m\}$ is a generalized processor system (GPS) iff $s_i(t) \ge s_{i+1}(t)$, $1 \le i < m$ for all t.

Theorem 1 [Sahni and Cho]: Let t_i, $1 \le i \le n$ be n job times such that $t_1 \ge t_2 \ge \ldots \ge t_n$. Let $\{P_1, P_2, \ldots, P_m\}$ be a GPS and let $T_k = \sum_{i=1}^{k} t_i$, $1 \le k < m$ and $T_m = \sum_{i=1}^{n} t_i$ (if n < m, then set $T_{n+1} = T_{n+2} = \ldots = T_m$). The given n jobs can be scheduled on the given GPS to finish by time d iff:

$$T_i \le \sum_{j=1}^{i} \int_{0}^{d} s_j(t)dt, \; 1 \le i \le m. \qquad []$$

One can prove that the preceding theorem is true even if the restriction that $s_i(t)$ is a nondecreasing function of t is removed. (Actually, Sahni and Cho's proof in [13] does not use the property that $s_i(t)$ is nondecreasing.) This observation will be implicitly used in Section 3.

2 m = 2

Without loss of generality, assume that $\mu_1 \ge \mu_2$ and $m_i \le \mu_1$, $1 \le i \le n$. Let n_1 be such that $m_i > \mu_2$, $1 \le i \le n_1$, and either $n_1 = n$ or $m_i \le \mu_2$, $n_1 < i \le n$ (it may require a job reordering to guarantee such a n_1). So, jobs 1, 2, ..., n_1 must be scheduled on P_1 while jobs n_1+1, n_1+2, ..., n may be scheduled on either P_1 or P_2. Further, assume that $t_{n_1+1} \ge t_i$, $n_1 + 2 \le i \le n$. We first observe that if there is a schedule S for the n jobs such that $C_{\max}(S) = d$, then there is a schedule with the same finish time d in which jobs 1, 2, ..., n_1 are scheduled on P_1 from time 0 to time $\tau = (\sum_{i=1}^{n_1} t_i)/s_1$ (see Figure 1).

To see this, note that the schedule S can be divided into intervals with the property that in each interval the jobs scheduled on P_1 are either all from $\{1, 2, \ldots, n_1\}$ or all from $\{n_1+1, n_1+2, \ldots, n\}$. Once this division into intervals has been done, we can easily construct, by interchanging intervals, a new schedule, S', in which all intervals containing jobs from $\{1, 2, \ldots, n_1\}$ are at the left. S' has a finish time d.

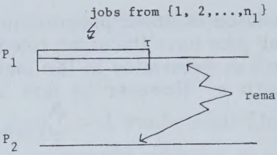
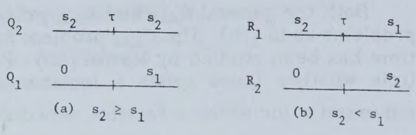


Figure 1



(a) $s_2 \geq s_1$ (b) $s_2 < s_1$

Figure 2 Forming the GPS

Hence, the $C_{\max}$ problem when m = 2 reduces to finding a minimum finish time schedule for jobs $\{n_1+1, \ldots, n\}$ on a two processor system $\{Q_1, Q_2\}$ in which Q_1 operates at speed 0 from time 0 to τ and at speed s_1 after τ; and Q_2 operates at speed s_2 from time 0. Furthermore, we may assume that Q_1 and Q_2 have the same memory size. If $s_2 \geq s_1$, then $\{Q_2, Q_1\}$ forms a GPS. If $s_2 < s_1$, then a GPS $\{R_1, R_2\}$ equivalent to $\{Q_1, Q_2\}$ may be formed by assigning to R_1 a speed of s_2 from 0 to τ and s_1 from τ on; R_2 has a speed of 0 from 0 to τ and s_2 from τ on (see Figure 2).

From Theorem 1, we see that the jobs $n_1+1, n_1+2, \ldots$, n can be scheduled to complete by d on the resulting GPS iff

$$t_{n_1+1} \leq s_2\tau + \max\{s_1, s_2\}(d - \tau)$$

and

$$\sum_{i=n_1+1}^{n} t_i \leq s_2 d + s_1(d - \tau)$$

Rearranging terms and combining, we see that d must satisfy the inequality:

$$d \geq \max\{\frac{t_{n_1+1} - s_2\tau}{\max\{s_1, s_2\}} + \tau, \frac{T_2 + s_1\tau}{s_1 + s_2}\}$$

where $T_2 = \sum_{i=n_1+1}^{n} t_i$. Let $T_1 = \sum_{i=1}^{n_1} t_i$. Substituting $\tau = T_1/s_1$ and adding the requirement that $d \geq \tau$, we obtain:

$$d \geq \max\{\frac{T_1}{s_1}, \frac{t_{n_1+1} - s_2 T_1/s_1}{\max\{s_1, s_2\}} + \frac{T_1}{s_1}, \frac{T_1 + T_2}{s_1 + s_2}\} \tag{1}$$

We immediately see that the minimum $C_{\max}$ is given by the right hand side of (1). An O(n) algorithm for the $C_{\max}$ problem is now easily obtained.

3 m = 3

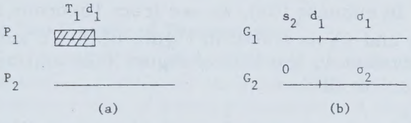
Assume that $\mu_1 \geq \mu_2 \geq \mu_3$ and that the jobs are ordered such that $m_1 \geq m_2 \geq \ldots \geq m_n$ (without loss of generality, we may assume that $m_i \in \{\mu_1, \mu_2, \mu_3\}$, $1 \leq i \leq n$ and so it takes only O(n) time to order the jobs). Let n_1 and n_2 be such that jobs 1, 2, ..., n_1 are the only jobs that cannot be processed on P_2 or P_3 because of memory requirements and jobs 1, 2, ..., n_2 are the only jobs that cannot be processed on P_3 because of memory requirements. Clearly, $n_1 \leq n_2$. For convenience, we now define $t_{n+1} = t_{n+2} = 0$. Let $T_1 = \sum_{i=1}^{n_1} t_i$ and $T_2 = \sum_{n_1+1}^{n_2} t_i$. Further, assume that $t_{n_1+1} \geq t_i$, $n_1+2 \leq i \leq n_2$ and $t_{n_2+1} \geq t_{n_2+2} \geq t_i$, $n_2+3 \leq i \leq n$.

A finish time d is said to be *feasible* iff there exists a schedule S for the given n jobs such that $C_{\max}(S) \leq d$. We shall proceed to obtain a set of necessary and sufficient conditions for d to be feasible. Once this has been done, we shall see how to find a minimum d satisfying these conditions and to then construct a schedule with minimum finish time. First, we see that d must satisfy the inequalities of (2) below:

$$\sum_{i=1}^{k} T_i \Big/ \sum_{i=1}^{k} s_i \leq d, \; k = 1, 2, 3 \tag{2}$$

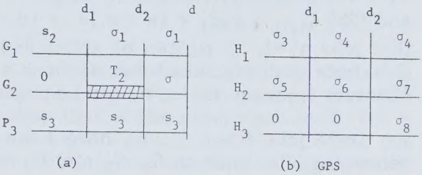
For the same reasons as in Section 2, we may limit our discussion to schedules in which jobs 1, 2, ..., n_1 are scheduled from 0 to d_1, $d_1 = T_1/s_1$ on P_1 (Figure 3(a)). Hence, the scheduling of n_1+1, ..., n_2 must be done on the GPS of Figure 3(b). In this figure, $\sigma_1 = \max\{s_1, s_2\}$ and $\sigma_2 = \min\{s_1, s_2\}$.

Let P_{fast} and P_{slow}, respectively, denote the processor with speed σ_1 and σ_2 (if $\sigma_1 = \sigma_2$, then fast = 1 and slow = 2). Let $C_1 = s_2 d_1$, and $C_2 = \sigma_1(d - d_1)$. Hence, C_1 is P_2's contribution to G_1 from 0 to d_1 while C_2 is P_{fast}'s contribution to G_1 from d_1 to d. Thus the total available processing capability on G_1 is $C_1 + C_2$. Let $C_3 = \sigma_2(d_2 - d_1)$ be the available processing on G_2.



T1 d1
P1
P2
(a)
s2 d1 σ1
G1
G2
0
σ2
(b)

Figure 3



d1 d2 d
s2 σ1 σ1
G1
G2
P3
0
T2
σ
s3 s3 s3
(a)
d1 d2
H1
H2
H3
σ3 σ4 σ4
σ5 σ6 σ7
0 0 σ8
(b) GPS

Figure 4

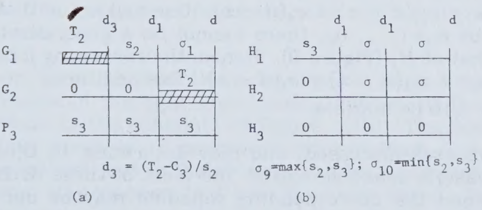
We shall consider three cases for T_2: (a) $T_2 \leq C_3$; (b) $C_3 < T_2 \leq C_1 + C_3$; and (c) $C_1 + C_3 < T_2$. When $T_2 \leq C_3$, a possible assignment for jobs n_1+1, ..., n_2 is to schedule them on G_2 from d_1 to d_2, $d_2 = d_1 + T_2/\sigma_2$ (Figure 4(a)). If this is done, the GPS that remains for the remaining jobs is as in Figure 4(b) where $\sigma_3 = \max\{s_2, s_3\}$; $\sigma_4 = \max\{\sigma_1, s_3\} = \max\{s_1, s_2, s_3\}$; $\sigma_5 = \min\{s_2, s_3\}$; $\sigma_6 = \min\{\sigma_1, s_3\}$; $\sigma_7 = \max\{\sigma_2, \sigma_6\}$; and $\sigma_8 = \min\{\sigma_2, \sigma_6\} = \min\{s_1, s_2, s_3\}$. Let $r_1(t)$ and $r_2(t)$, respectively, denote the speed of H_1 and H_2 at time t, $0 \leq t \leq d$. We immediately see that there is no sceduling of jobs n_1+1, ..., n_2 that yields a larger value for $\int_0^d r_1(t)dt$ or $\int_0^d r_1(t)dt + \int_0^d r_2(t)dt$. Hence, from Theorem 1, it follows that (2) together with the following ((3) and (4)) form a set of necessary and sufficient conditions for d when $T_2 \leq C_3$:

$$\begin{aligned} t_{n_2+1} &\leq \sigma_3 d_1 + \sigma_4(d - d_1) \\ &= d_1\max\{s_2,s_3\} + (d-d_1)\max\{s_1,s_2,s_3\} \end{aligned} \tag{3}$$

$$\begin{aligned} t_{n_2+1} + t_{n_2+2} &\leq (\sigma_3+\sigma_5)d_1 + (\sigma_4+\sigma_6)(d_2-d_1) + (\sigma_4+\sigma_7)(d-d_2) \\ &= d_1(s_2+s_3) + (d_2-d_1)(s_3+\sigma_1) \\ &\quad + (d-d_2)(s_1+s_2+s_3-\min\{s_1,s_2,s_3\}) \end{aligned} \tag{4}$$

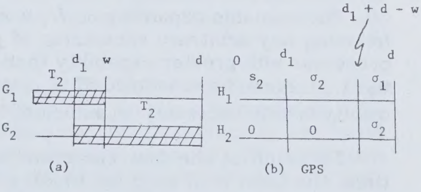
Next, consider the case $C_3 < T_2 \leq C_1 + C_3$. If we schedule the jobs n_1+1, ..., n_2 on G_2 from d_1 to d and on G_1 from 0 to $(T_2 - C_3)/s_2$ (Figure 5(a)), then we will be left with the GPS of Figure 5(b). The processing capability, K, of H_1 is seen to be :

$$\begin{aligned} K &= s_3(T_2 - C_3)/s_2 + (d_1 - (T_2 - C_3)/s_2)\max\{s_2, s_3\} \\ &\quad + (d - d_1)\max\{s_1, s_2, s_3\} \end{aligned}$$



d3 d1 d
T2
s2 σ1
G1
G2
P3
0 0
T2
s3 s3 s3
d3 = (T2-C3)/s2
(a)
d1 d1 d
s3 σ σ
H1
H2
H3
0 σ σ
0 0 0
σ9=max{s2,s3}; σ10=min{s2,s3}
(b)

Figure 5



d1 w d
T2
G1
G2
T2
(a)
d1 + d - w
0 d1 d
s2 σ2 σ1
H1
H2
0 0 σ2
(b) GPS

Figure 6

We also see that no matter how jobs $n_1+1, \ldots, n_2$ are scheduled on G_1 and G_2, we cannot create a generalized processor from the remaining idle times on G_1, G_2, and P_3 such that its procesing capability from 0 to d exceeds K. The total processing capability remaining on G_1, G_2, and G_3 will of course be the same no matter how jobs $n_1+1, \ldots, n_2$ are scheduled. From these observations and Theorem 1, we see that if (2) is satisfied and $C_3 < T_2 \leq C_1 + C_3$ then it is both necessary and sufficient that $t_{n_2+1} \leq K$ in order for d to be feasible.

Now, let us consider the final case, $T_2 > C_1 + C_3$. Let $w = d_1 + (T_2 - C_1 - C_3)/\sigma_1$. From (2), it follows that $T_2 \leq C_1 + C_2 + C_3$. This together with $T_2 > C_1 + C_3$ implies that $d_1 < w \leq d$. We shall consider the two subcases: (3a) $t_{n_1+1} \leq s_2 d_1 + (w - d_1)\sigma_1 + (d - w)\sigma_2$ and (3b) $t_{n_1+1} > s_2 d_1 + (w - d_1)\sigma_1 + (d - w)\sigma_2$. In subcase (3a), we see from Theorem 1 that jobs $n_1+1, \ldots, n_2$ can be scheduled on G_1 and G_2 as shown in Figure 6(a). To see this, note that the scheduled portions are equivalent to the GPS of Figure 6(b) and by Theorem 1, jobs $n_1+1, \ldots, n_2$ can be scheduled on this GPS.

Once jobs $n_1+1, \ldots, n_2$ have been scheduled on G_1 and G_2 in this manner, the remaining idle times on G_1, G_2, and P_3 yield the GPS of Figure 7. From Theorem 1, it follows that jobs $n_2+1, \ldots,$ n can be scheduled on the GPS of Figure 7 iff $t_{n_2+1} \leq s_3 w + \sigma_4(d - w)$.

In addition, we note that no matter how jobs $n_1+1, \ldots, n_2$ are scheduled on G_1 and G_2, the remaining idle times on G_1, G_2, and P_3 cannot yield a generalized processor with capability more than $s_3 w + \sigma_4(d - w)$. Hence, when (2) holds, $T_2 > C_1 + C_3$ and $t_{n_1+1} \leq s_2 d_1 + (w - d_1)\sigma_1 + (d - w)\sigma_2$, then it is both necessary and sufficient that $t_{n_2+1} \leq s_3 w + \sigma_4(d - w)$ in order for d to be feasible.

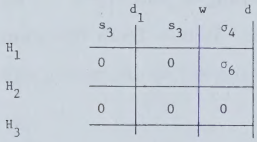


Figure 7

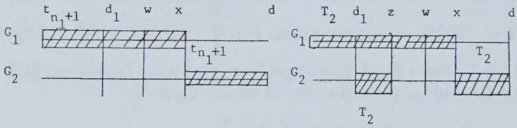


Figure 8

For subcase (3b), we have $t_{n_1+1} > s_2 d_1 + (w - d_1)\sigma_1 + (d - w)\sigma_2$ (in addition to (2) and $T_2 > C_1 + C_3$). First, we observe that it must be the case that $t_{n_1+1} \leq C_1 + C_2$ as otherwise, by Theorem 1, t_{n_1+1} cannot be scheduled on G_1 and G_2. Let x be such that $t_{n_1+1} = C_1 + (x - d_1)\sigma_1 + (d - x)\sigma_2$. I.e., $x = d_1 + (t_{n_1+1} - C_1 - C_3)/(\sigma_1 - \sigma_2)$ (note that $\sigma_1 > \sigma_2$ when $s_2 d_1 + (w - d_1)\sigma_1 + (d - w)\sigma_2 < t_{n_1+1} \leq C_1 + C_2$). Clearly, $w < x \leq d$ and t_{n_1+1} can be scheduled on G_1 and G_2 as in Figure 8(a).

Let $z = d_1 + (T_2 - t_{n_1+1})/\sigma_2$. From our earlier definitions of w and x, it follows that $z < w < x$. Jobs $n_1+2, \ldots, n_2$ may be scheduled on G_2 from d_1 to z (Figure 8(b)). Once this is done, the idle times on G_1, G_2, and P_3 are equivalent to the GPS of Figure 9. In Figure 9, $\sigma_{11} = \max\{\sigma_2, s_3\}$ and $\sigma_{12} = \min\{\sigma_2, s_3\}$.

The available capability on H_1 is $s_3 z + \sigma_{11}(x - z) + \sigma_4(d - x)$. One can argue that following any arbitrary scheduling of jobs $n_1+1, \ldots, n_2$, there cannot be a generalized processor with greater capability than that of H_1 (Figure 9). Hence, the remaining jobs $n_2+1, \ldots,$ n can be scheduled iff $t_{n_2+1} \leq s_3 z + \sigma_{11}(x - z) + \sigma_4(d - x)$. This additional inequality is both necessary an sufficient for d to be feasible.

For each of the four cases and subcases discussed, one may determine, in O(n) time, the least d (if any) for which a feasible schedule exists. The least of these ds is C_{max}. Once the C_{max} has been determined the corresponding schedule may be constructed in O(n) time as discussed.

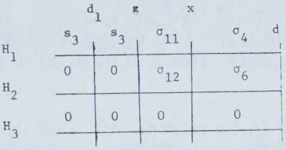


Figure 9

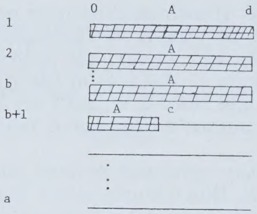


Figure 10 Scheduling R on A

4 Two Classes of Processors

Let A and B be two classes of processors such that all processors in A have the same speed, s_A, while all in B have the same memory size, μ_B. Further, assume that the memory size of every processor in A is greater than that of the processors in B. The n jobs to be scheduled may be partioned into two disjoint sets R and T such that the memory requirement of every job in R exceeds μ_B while the memory requirement of every job in T is less than or equal to μ_B.

It is clear that jobs in R can be scheduled only on processors in A. The set of processors in A forms a system of identical processors with memory. Kafura and Shen [8] have developed necessary and sufficient conditions for d to be feasible for a system of identical processors with memory. Let |A| = a and |R| = r. Assume that $\mu_1 \ge \mu_2 \ge \dots \ge \mu_a$ and that $m_1 \ge m_2 \ge \dots \ge m_r$. Let F_i denote the subset of R consisting only of those jobs with memory requirement more than μ_{i+1}, $1 \le i < a$. Let F_a = R. We assume that $m_1 \le \mu_1$. Let X_i be the sum of the processing times of all jobs in F_i, $1 \le i \le a$. Kafura and Shen [8] have shown that the job set R can be scheduled on the processor set A to complete by time d iff d satisfies the inequality:

$$d \ge f^* = \max\{ \max_{1 \le i \le r}\{t_i / s_A\}, \max_{1 \le i \le a}\{X_i / (s_A i)\} \} \tag{17}$$

Furthermore, when d satisfies the above inequality the desired schedule may be obtained by scheduling the jobs in the order 1, 2, ..., r using McNaughton's strategy [12] on the processors in the order 1, 2, ..., a. The resulting schedule fully utilizes processors 1, 2, ..., b where b = $\lfloor \sum_{i=1}^{r} t_i / (s_A d) \rfloor$. In addition, if b < a then processor b+1 is utilized from 0 to c = rem($\sum_{i=1}^{r} t_i$, $s_A d$) / s_A where rem(x, y) is the remainder of x divided by y. Hence the schedule is as in Figure 10.

Let S be any schedule for the jobs in R ∪ T. Let $C_{\max}(S)$ = d. Clearly, d ≥ f*. We shall show that S can be transformed into a schedule Q with $C_{\max}(Q)$ = d and Q is such that the jobs in R are scheduled as in Figure 10. Suppose that the jobs in R are not scheduled as in Figure 10. Let Δ be such that d/Δ is an integer and there are no preemptions or completion in the interval (iΔ, (i+1)Δ), 0 ≤ i < d/Δ. Since the number of preemptions in S is finite and since all the values concerned are rationals, such a Δ clearly exists. Note that, by definition of Δ, it is not possible for two jobs to be scheduled in the same Δ interval on any processor. Figure 11(a) shows a possible configuration for S. The heavy lines represent jobs in R; the light lines represent either jobs in T or idle times.

First, we shift the heavy lines in each Δ interval to the top of the Δ interval. This results in the light lines moving downwards (but remaining on the A processors). The result is the schedule of Figure 11(b). This shifting preserves feasibility as the heavy lines move up to processors with at least as much memory capacity. The light lines represent jobs in T and these can be processed on any of the processors in A. In addition, all procesors in A have the same speed. So, the amount of processing done in a Δ interval is the same for all procesors in A.

The next transformation is to permute the Δ intervals so that the number of heavy lines in each interval is nonincreasing , left to right. This results in the schedule of Figure 11(c). As we can see, our schedule is still not in the form of that in Figure 10. In order to obtain a schedule in the right form, slots a, b, and c need to be replaced by heavy lines and slots d, e, and f are to be replaced by light lines. This can be done in a systematic manner as discussed below.

Select the rightmost column into which a heavy line is to be introduced. Call this column u. In this column select the uppermost slot containing a light line. In Figure 11(c), this results in slot a being selected. Next, select the leftmost column into which a light line is to be introduced. Call this column v. In this column, select the lowest slot into which there is a heavy line. This yields slot f of Figure 11(c). It should be clear that column u has fewer heavy lines in it than does column v. Hence, at least one of the heavy lines in v represents a job not scheduled in column u. We shall select one such heavy line to be moved into column u. This selection is done by first lining up heavy lines representing jobs common to u and v.

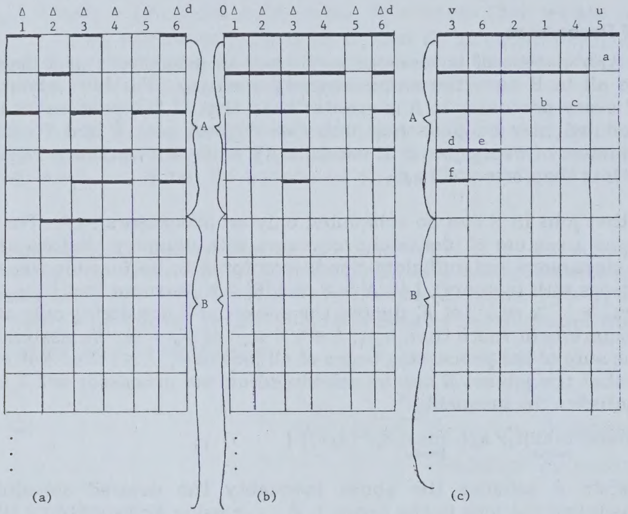
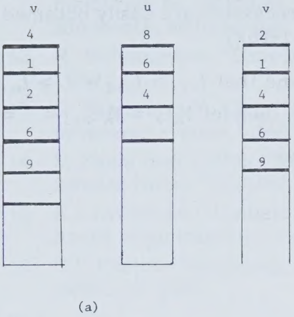


Figure 11

Suppose the job representation for heavy lines for an example u and v are as in Figure 12(a). Suppose job i is in both u and v and that it is in row r_u of u and row r_v of v. If $r_u < r_v$, then i in column u is interchanged with the line in row r_v of column u. This preserves feasibility as $m_i \leq \mu_{r_v}$. If $r_u > r_v$, then i in column v is interchanged with the line in row r_u of column v. This also preserves feasibility. When these interchanges are performed on Figure 12(a), the configuration of Figure 12(b) is obtained.

For movement into column u, we may select any one of the heavy lines in column v that are lined up with a light line in column u (lines 1 and 9 in Figure 12(b)). This guarantees that the new set of heavy lines in column u can be feasibly assigned to the processors. The remaining heavy lines in v may be shifted upwards.

Now, we have to move one of the light lines in u out to column v. Suppose that u and v are as in Figure 13(a). X_1 denotes the jobs from T assigned to rows of u that have light lines in u but heavy ones in v. Let the number of heavy lines in v be w. From the choice of u and v, it follows that $|X_1| = z \geq 1$. If X_1 contains a job that has not been scheduled in column v, then this job may be selected for movement into column v. Otherwise, permute processors w+1, ..., m so that the set of jobs scheduled in column v on processors w+1, w+2, ..., w+z is X_1. Let X_2 denote the set of jobs scheduled in u on processors w+1, ..., w+z. If every job in X_2 is scheduled in column v, then permute processors w+z+1, ..., m so that the set of jobs scheduled on w+z+1, ..., w+2z in column v is X_2. This permuting of processors is continued until we have an X_k such that at least one job in X_k is not scheduled in column v. Such an X_k exists as u has fewer heavy lines than does v. Let $x_0 \in X_k$ be such that x_0 is not scheduled in v. Interchange x_0 with the job, x_1, scheduled on the same processor in v. Clearly, $x_1 \in X_{k-1}$. Hence x_1 is scheduled in u on one of the processors w + (k-3)z + j, $1 \le j \le z$. Let this processor have index q. Interchange the assignment of x_1 on processor q column u with that of the job, x_2, in column v of q. Now, $x_2 \in X_{k-2}$ and x_2 is scheduled on q, in column u for some q, $q \in \{w+(k-4)z+j \mid 1 \le j \le z\}$. We may continue interchanges in this way until we determine $x_{k-1} \in X_1$ that needs to be moved out of X_1 from column u. This x_{k-1} is to be moved into the last heavy line spot in column v.


v
4
1
2
6
9
u
8
6
4
v
2
1
4
6
9
u
8
4
6

(a) (b)

Figure 12

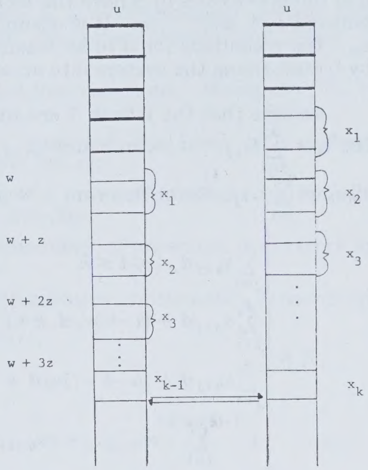
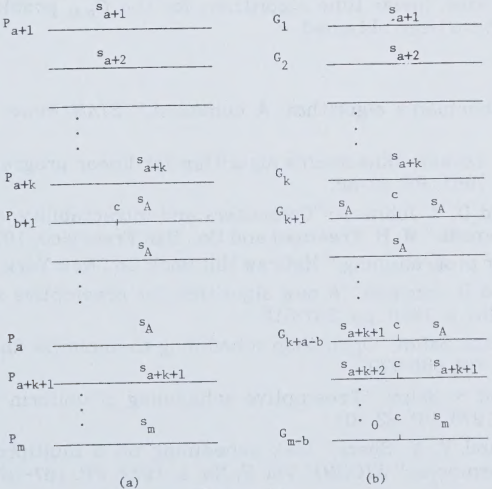

u
u
w
w + z
w + 2z
w + 3z
x_1
x_2
x_3
x_{k-1}
x_1
x_2
x_3
x_k

Figure 13


P_{a+1}
s_{a+1}
s_{a+2}
P_{a+k}
s_{a+k}
P_{b+1}
c
s_A
s_A
P_a
s_A
P_{a+k+1}
s_{a+k+1}
P_m
s_m
G_1
s_{a+1}
G_2
s_{a+2}
G_k
s_{a+k}
G_{k+1}
s_A
c
s_A
s_A
G_{k+a-b}
s_{a+k+1}
c
s_A
s_{a+k+2}
c
s_{a+k+1}
G_{m-b}
0
c
s_m

(a) (b)

Figure 14

By repeating the interchange process described above a finite number of times, the configuration of Figure 11(c) can be transformed into the form of Figure 10. Having established this result, we see that we need concern ourselves only with schedules in which jobs in R are scheduled as in Figure 10. For any d, the remaining idle times in A and the processors in B have the form given in Figure 14(a). The processors from B are indexed a+1, a+2, ..., m. It is assumed that $s_{a+1} \geq s_{a+2} \geq \ldots \geq s_{a+k} \geq s_A \geq s_{a+k+1} \geq \ldots \geq s_m$. The conditions for d to be feasible on this system of processors are easily obtained by transforming the system into an equivalent GPS (Figure 14(b)).

Assume that the jobs in T are indexed r+1, r+2, ..., n and that $t_{r+1} \geq t_{r+2} \geq \ldots \geq t_n$. Let $V_i = \sum_{j=1}^{i} t_{r+j}$, $1 \leq$ i < m-b and $V_{m-b} = \sum_{j=1}^{n-r} t_{r+j}$. If n-r < m-b, then let $V_{n-r} = V_{n-r+1} = \ldots = V_{m-b} = \sum_{j=1}^{n-r} t_{r+j}$. From Theorem 1, we see that d is feasible iff:

$$V_i \leq \begin{cases} \sum_{j=1}^{i} s_{a+j} d, \; 1 \leq i \leq k \\ \sum_{j=1}^{k} s_{a+j} d + (i-k) s_A d, \; k+1 \leq i < k+a-b \\ \sum_{j=1}^{k} s_{a+j} d + (a-b-1) s_A d + s_A (d-c) \\ + \sum_{j=1}^{i-(k+a-b)} d s_{a+k+j} + c s_{b+i+1}, \; k+a-b \leq i \leq m-b \end{cases} \tag{18}$$

For a given b, the least d (if any) that satisfies (17) and (18) can be found in O(n) time. Since as b decreases (increases) this least d increases (decreases), the optimal b can be obtained in O(log a) = O(log m) attempts. Hence, the least feasible d can be obtained in O(n log m) time. Note that the $X_i s$ (Eq (17)) can be obtained in O(r log a) time and that $V_i s$ (Eq (18)) can be obtained in O((n-r) + (m-b) log (m-b)) time (as only the largest m-b jobs in T need to be ordered). So, the total time needed to compute $(C_{\max})_{\min}$ is O(n log m) (assume n $\geq$ m). The actual schedule can be constructed in an additional O(mn) time using the algorithm of [12] for R and that of [13] for T. The time O(mn) can be reduced to O(n) using a scheme similar to that of [7] for T.

5. Conclusions

For three special cases, linear time algorithms for the $C_{\max}$ problem when all jobs are released at time 0 have been obtained.

References

1. G. Dantzig, "Khachian's algorithm: A comment," *SIAM News*, Vol. 13, No. 5, Oct. 1980.
2. P. Gacs and L. Lovasz, "Khachian's algorithm for linear programming," *Math. Prog. Study*, Vol. 14, 1981, PP. 61-68.
3. M. R. Garey and D. S. Johnson, "Computers and intractability, a guide to the theory of NP-Completeness," W. H. Freeman and Co., San Francisco, 1979.
4. S. Gass, "Linear programming," McGraw Hill Book Co., New York, 1969.
5. T. Gonzalez and D. Johnson, "A new algorithm for preemptive scheduling of trees," *JACM*, Vol. 27, No. 2, 1980, pp. 287-312.
6. T. Gonzalez and S. Sahni, "Open shop scheduling to minimize finish time," *JACM*, Vol. 23, No. 4, 1976, PP. 665-679.
7. T. Gonzalez and S. Sahni, "Preemptive scheduling of uniform processor systems," *JACM*, Vol. 25, 1978, PP. 92-101.
8. D. G. Kafura and V. Y. Shen, "Task scheduling on a multiprocessor system with independent memories," *SICOMP*, Vol. 6, No. 1, 1977, PP. 167-187.

9. L. G. Khachian, "A polynomial algorithm in linear programming," *Soviet Math. Dokl.*, Vol. 20, No. 1, 1979, PP. 191-194.

10. V. Klee and G. L. Minty, "How good is the simplex algorithm?" in O. Shisha (ed.) *Inequalities III*, Academic Press, New York, 1972, PP. 159-175.

11. T. H. Lai and S. Sahni, "Preemptive scheduling of a multiprocessor system with memories to minimize $L_{\max}$," Report No. 81-20, Computer Science Dept., Univ. of Minnesota, Minneapolis, 1981.

12. R. McNaughton, "Scheduling with deadlines and loss functions," *Manag-Sci*, 12, 7, 1959.

13. S. Sahni and Y. Cho, "Scheduling independent tasks with due times on a uniform processor system," *JACM*, Vol. 27, No. 3, 1980, PP. 550-563.

14. S. Sahni and Y. Cho, "Nearly on line scheduling of a uniform processor system with release times," *SICOMP*, Vol. 8, No. 2, 1979, pp. 275-285.

15 E.L.Lawler and J.Labetoulle, "On preemptive scheduling of unrelated processors by linear programming", J.ACM, 25, 612-619, 1978.

16. C.U.Martel, "Scheduling multiple processors with memory constraints", Proceedings IMACS 10, 1982.

DECENTRALIZED ESTIMATORS FOR LARGE GENERATION/LOAD DISTURBANCES IN INTERCONNECTED POWER SYSTEMS

N. ČUKALEVSKI
Mihailo Pupin Institute
Belgrade, Yugoslavia

M. ĆALOVIĆ
Department of Electrical Engineering
University of Belgrade, Yugoslavia

J. MEDANIĆ
Department of General Engineering and Coordinated Science
Laboratory, University of Illinois, and
Mihailo Pupin Institute, Belgrade, Yugoslavia

ABSTRACT

The problem of fast detection of large power unbalances in interconnected power systems by low order dynamic observers is considered. The proposed detector-identifier is constructed based on a time-decomposition of the relevant system dynamics and estimates the power mismatch using only local measurements. Because the problem is typical of other interconnected systems the solution is derived for a general interconnected system and the results are then specialized to the case of an interconnected power system.

INTRODUCTION

Control of interconnected power systems in the emergency state when large power mismatches between generation and load cause a viability crisis [1] is an important aspect of power system control. The main control objective in such situation is to prevent the transition into a disintegrative state [1] where certain interconnections may be severed and further structural perturbations in the system configuration may occur. Presently the main operational strategy for arresting viability crises rests on load shedding (or starting fast units in the opposite case), with load shedding apparatus set to become operative at selected frequency deviations. Alternatively, load shedding may be done centrally, from the dispatching center. In both cases the levels of load shed are selected to offset the power mismatch which, however, is not measured.

The main practical problem is the determination, in a sufficiently short interval of time, of the initial magnitude of power mismatch, as well as the power area where the mismatch occurred. Availability of such an estimate would enable the implementation of fast load shedding strategies, in full agreement with the area control error concept [3,4] presently used in interconnected power systems to carry out normal power balance adjustments necessary due to daily variations of the load.

This paper addresses the problem of fast detection of power mismatches in interconnected power systems and indicates how low order observers can be employed within the framework of the present operational practice based on the area control error concept for fast detection of the level and area of origin of the power mismatch utilizing only locally available information. The area control error concept is seen, however, as a significant concept for large scale interconnected systems in general, and so a problem formulation applicable to large scale systems in general is presented first. A coordinated procedure for compensating large constant disturbances generalizing the area control error concept is proposed. It is shown next that local low order observers can be devised to detect in each subsystem the presence and the level of large disturbances based only on the locally available information. Finally, the results are specialized to the power system problem and observers for fast detection of the area control error in interconnected power systems are determined.

FORMULATION OF THE PROBLEM

Consider the interconnected system

$$\dot{x}_i = A_{ii}x_i + \sum_{j\neq i}^{k} d_{ij}z_{ij} + g_iv_i + b_iu_i$$

$$\dot{z}_{ij} = p_{ij}x_j + p_{ji}x_i$$

$$y_i = c_ix_i \tag{1}$$

where $x_i \in R^{n_i}$ and where, for simplicity, it is assumed that u_i is a scalar control, v_i is a scalar disturbance, y_i is a scalar output measured and regulated by the i-th subsystem, z_{ij} is a scalar interaction variable between subsystem i and j, and all matrices and vectors are of appropriate dimension. Assume each subsystem has available the local measurement, as well as the measurements of all interaction variables connecting it to other subsystems. Thus, the information structure is defined by the following outputs being available to the i-th decision maker:

$$y_i = C_ix_i$$

$$y_{ij} = z_{ij}, \qquad j = 1,\ldots,k, \quad j\neq i. \tag{2}$$

Model (1) is assumed to represent a linearized representation of the interconnected system about a reference operating point. Steady state operation is defined as holding the outputs and interaction variables at desired values, so state variables and outputs in (1) represent deviations from nominal values and should be regulated to zero. The disturbances v_i in (1) are also deviations from expected external inputs and in normal operation are considered to be mutually uncorrelated stochastic variables with zero mean and small covariances. The controls are specified functions of the available measurements designed to insure desired dynamic response to stochastic perturbations in the v_i and to regulate y_i under normal operating conditions.

When large lasting perturbations in some v_i occur, these are sensed throughout the system via deviations in the y_i and y_{ij}, and while excursions in the y_i are regulated those of the y_{ij} are not. There are many reasons why the perturbations should be detected and eliminated as soon as possible. First, the unbalance caused by these perturbations may drive the system far from the nominal operating point and performance expected from normal controls may no longer follow. Second, large excursions from the reference point may result in violation of certain constraints. As a consequence severance of certain system connections may follow, possibly affecting the system's integrity. Also, interconnection of a subsystem with the other subsystems is formed to obtain cost effective and secure operation and generally has the consequence that deviations of subsystem states around the reference point are smaller than when the system is operating in isolation. This has the desirable consequence of expanding the operating conditions in which the linearized model of the subsystem is valid. This provides additional motivation to maintain the interconnection ties in the normal state. Finally, an unbalance usually has the effect of a net perturbed flow of some commodity from one subsystem to another. Since this flow has not been scheduled, cost is normally incurred by the system receiving the commodity which is greater than that which would be incurred for a scheduled transfer and is thus undesirable.

For the above reasons it is a desired goal of all subsystems to maintain normal operating conditions in the interconnection. The problem, then, is how to determine additional, supplementary controls to return the system to its normal operating state, defined as that where both regulated outputs and interaction variables are reduced to zero. Due to decentralized information structure uncoordinated action on the part of subsystem controllers may bring about a new unbalance, or the system may be balanced but far from the operating point because control action was applied in a different subsystem than where the unbalance originated. This will, in fact, always occur under the action of existing controls since all controls will strive to regulate the ouptuts ofsetting the flow through interaction variables. Thus, a coordinated strategy must be employed to prevent system hunting and possible disintegration due to lasting constraint violations.

Problems of this kind may be found in many large scale systems where certain commodities flow within the system for the purpose of achieving a more cost-effective and secure operation. Power systems are a typical example. The strategy concept proposed in the sequel reflects directly a long-standing practice in the operation of interconnected power systems, which should be recognized for its wider significance. At the same time, the problem of fast detection of unbalances in power systems and the proper execution of control actions to alleviate large unbalances are not resolved, and so, the application of results available in control to the power problem is of interest. The purpose of this paper is to define the supplementary control strategy concept for large scale systems in general, and then to show how results on observing constant disturbances via low-order observers can be used to detect unbalances using only locally available information. Finally, we indicate how the results can be used in power systems for fast detection of the load unbalance in interconnected power systems.

SUPPLEMENTARY CONTROLS

To restore the system to its normal operating state it is proposed that supplementary controls u_{is} be introduced so that the total controls are $u_i = u_{in} + u_{is}$. The component u_{in} provides the normal regulation around an operating point for small stochastic disturbances, and u_{is} provides supplementary controls used only when large lasting perturbations in v_i occur. To avoid uncoordinated application of supplementary controls which may result in hunting and overreaction, it is proposed that the following strategy be employed:

(i) All subsystems participate in the control of interconnected systems during unbalanced conditions using normal controls.

(ii) Local detection of unbalances is undertaken by all subsystems to determine the level and subsystem-or-origin of the disturbance.

(iii) Supplementary control is applied only in the subsystem where the unbalance originated.

When an unbalance occurs in the i-th subsystem the effect of (i) will be that all subsystems will participate in regulating the outputs by creating a reverse unbalance of necessary degree in their own subsystem. The effect of (iii) will be that when the i-th subsystem eliminates the unbalance in its subsystem using supplementary controls, other subsystems will return to the normal operating state existing before the unbalance occurred and lasting changes will persist only in the i-th subsystem.

Step (ii) is a prerequisite for realizing the proposed strategy. In this paper we consider the problem of detecting both the level and subsystem of origin of the disturbance under the assumptions that the disturbance is large enough and lasting so that application of supplementary controls is necessary and that the detection can be carried out sufficiently fast so that the disturbance can be considered constant in the detection interval. The significant result of this analysis is that the assumed information structure, which restricts the subsystems to local information only is not an impediment and that each subsystem can detect large perturbations in its own subsystem and thus carry out the proposed strategy.

OBSERVING A CONSTANT DISTURBANCE

In this section the basic structure of the observer needed to detect a constant disturbance is reviewed to highlight the issues related to the construction of low order, approximate, observers.

Consider an isolated subsystem modeled by

$$\begin{aligned} \dot{x} &= Ax + bu_d + gv \\ y &= cx \end{aligned} \tag{3}$$

where the subscript i has been dropped for convenience, and consider the problem of fast detection of the constant disturbance v. Appending v as the (n+1)-th state, the problem reduces to that of observing the component $z = x_{n+1}$ of the state given the expanded system

$$\dot{x}_e = \begin{bmatrix} A & g \\ 0 & 0 \end{bmatrix} x_e + \begin{bmatrix} b \\ 0 \end{bmatrix} u_d \tag{4}$$

$$\begin{aligned} y &= [c \quad 0]x_e \\ z &= [0_n \quad 1]x_e . \end{aligned} \tag{5}$$

This is, as is well-known, accomplished by an observer of structure

$$\dot{\bar{x}}_e = \left(\begin{bmatrix} A & g \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} H \\ h \end{bmatrix} [c \quad 0] \right) \bar{x}_e + \begin{bmatrix} b \\ 0 \end{bmatrix} u_d + \begin{bmatrix} H \\ h \end{bmatrix} y \tag{6}$$

$$z_e = [0 \quad 1]\bar{x}_e$$

where the observer gain $\begin{bmatrix} H \\ h \end{bmatrix}$ is selected to insure fast convergence of the estimate, i.e., to insure that the state error, defined by $c = x_e - \bar{x}_e$ and governed by

$$\dot{e} = \left(\begin{bmatrix} A & g \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} H \\ h \end{bmatrix} [c \quad 0] \right) e \tag{7}$$

decays sufficiently fast. Because the observation should be fast in the considered class of problems it is of interest to consider the consequences of using an approximate, reduced order observer of the form

$$\dot{x}_r = (A_r - H_r c_r) x_r + g_r z + H_r y + b_r u_n .$$

Incorporating the normal control u_n into the resulting state matrix, decompose to this effect the extended state vector as $x_e = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ and write (4)-(5) as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$y = [c_1 \quad c_2] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} , \qquad z = [0 \quad d_2] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{9}$$

where the dimension of x_2 equals the dimension selected for the reduced order observer to be constructed. By applying a block triangularizing transformation [8,9] (see Appendix II), the system (9) can be transformed to the form

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} F_{11} & 0 \\ A_{21} & F_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \begin{aligned} y &= c_1 x_1 + D_2 x_2 \\ z &= d_2 x_2 \end{aligned} \tag{10}$$

with z still a function of only x_2. The reduced order observer is now constructed as a replica of the second subsystem and has the form

$$\dot{x}_e = F_{22} x_e + H(y - y_e) , \qquad y_e = D_2 x_e \tag{11}$$

while the first subsystem, here called the residual subsystem, is not modeled in the observer. Defining the estimation error by $e = x_2 - x_e$, the error equation is found to have the form

$$\dot{e} = (F_{22} - HD_2) e + (A_{21} - Hc_2) x_1 . \tag{12}$$

From (12) it is clear that the transient behavior of the estimation error, and thus the speed of convergence, is influenced by the observer dynamics $(F_{22} - HD_2)$, and by the residual subsystem dynamics, because these dynamics appear as a forcing input in the error equation. Therefore, three important conclusions can be drawn:

(a) The spectrum of F_{22}, denoted by $S(F_{22})$, should contain all dominant eigenvalues (slow dynamics) of the system, because then $S(F_{11})$ will contain only the nondominant (fast) system eigenvalues, and thus the influence of the term dependent on x_1 in (12) will decay as fast as possible with the selected dimension of the reduced order observer. This means that the observer should model the slow dynamics of the system model, and the block triangularizing transformation selected should achieve a slow-fast decomposition of the system model.

(b) The observer dimension should be determined from the spectral portrait of the system matrix, i.e., from the number of slow eigenvalues in $S(\bar{A})$, with "slow" being a relative qualification dependent on the clusering of system eigenvalues.

(c) It is unnecessary for the observer spectrum, $S(F_{22}-HD_2)$, to be "much faster" than the spectrum of F_{11}, since if it is, the speed of convergence will be constrained by the speed of decay of x_1'.

EXTENSION TO INTERCONNECTED SYSTEMS

Consider now the interconnected system (1) and consider the problem of detecting the constant disturbance in the i-th subsystem. Appending the disturbance to the substate x_i, it is obtained in full analogy with the previous development

$$\dot{x}_{ie} = \begin{bmatrix} A & g \\ 0 & 0 \end{bmatrix} x_{ie} + \sum_{j\neq i} \begin{bmatrix} d_{ij} \\ 0 \end{bmatrix} z_{ij} + \begin{bmatrix} b_i \\ 0 \end{bmatrix} u_i \tag{13}$$

$$y_i = [c_i \quad 0]x_{ie}$$

$$z_i = [0 \quad 1]x_{ie}.$$

Let now the observer be constructed in the form

$$\dot{\bar{x}}_{ie} = \left(\begin{bmatrix} A & g \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} H_i \\ h_i \end{bmatrix} [c_i \quad 0] \right) \bar{x}_{ie} + \begin{bmatrix} b_i \\ 0 \end{bmatrix} u_{id} + \sum_{j\neq i} \begin{bmatrix} d_{ij} \\ 0 \end{bmatrix} z_{ij} + \begin{bmatrix} H_i \\ h_i \end{bmatrix} y \tag{14}$$

$$z_{ie} = [0 \quad 1]\bar{x}_{ie}.$$

It follows that the error in the estimate of x_{ie}, denoted by $e_i = x_{ie} - \bar{x}_{ie}$, is governed by the error equation

$$\dot{e}_i = \left(\begin{bmatrix} A & g \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} H_i \\ h_i \end{bmatrix} [c_i \quad 0] \right) e_i \tag{15}$$

and thus with proper choice of $H_i \in R^{n\times 1}$ and h_i, this observer can be used to detect the perturbations in v_i. Consider now observation by a reduced order observer. Decomposing the extended substate x_{ie} as $\begin{bmatrix} x_1^i \\ x_2^i \end{bmatrix}$ with x_2^i the dimension of the reduced order observer, it is easily obtained after application of a suitable transformation that (13) reduces to

$$\begin{bmatrix}\dot{x}_1'\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}F_{11} & 0\\ A_{21} & F_{22}\end{bmatrix}\begin{bmatrix}x_1'\\ x_2\end{bmatrix} + \begin{bmatrix}b_1+Kb_2\\ b_2\end{bmatrix}u_{id} + \Sigma\begin{bmatrix}d^1_{ij}+Kd^2_{ij}\\ d^2_{ij}\end{bmatrix}z_{ij} \qquad (16)$$

$$y_i = [c_1 \quad (c_2-c_1K)]\begin{bmatrix}x_1'\\ x_2\end{bmatrix} = c_2x_2$$

$$z_i = d_2x_2.$$

The reduced order observer is then constructed to mimic the second subsystem in the form

$$\begin{aligned}\bar{x}_r &= F_{22}\bar{x}_r + b_2u_{id} + \Sigma d^2_{ij}z_{ij} + H_i(y-y_e)\\ y_e &= c_2\bar{x}_r\\ \bar{z}_e &= d_2\bar{x}_r.\end{aligned} \qquad (17)$$

Defining the estimation error as $e = x_2 - \bar{x}_r$ the error equation is found to have the form

$$\dot{e}_i = (F_{22}-H_id_2)e + (A_{21}-H_ic_1)x_1'. \qquad (18)$$

Clearly, the same conclusions apply as in the case of isolated subsystems. Thus the reduced order observer that captures the slow dynamics of the subsystem dynamics can be used to estimate unbalance in the subsystem using only locally available information.

It is noted that the detection of constant disturbances in the interconnected system based on locally available information is possible because all ineraction variables are, by assumption, measured. (This will, in fact, be the case in many problems because the interaction variables represent some commodity flowing from one subsystem to another.) It is also to be noted that large perturbations in interconnected systems may result in significant departures from the operating point and in structural perturbations, invalidating in any case the linear model used for the subsystem. The affected subsystem needs in this extreme case to also identify the proper subsystem model, but all other subsystems may continue to employ linear models to determine the correct estimates. In comparison, an observer depending on the information from the entire system would not be applicable whenever such a disturbance occurred in any subsystem within the interconnected system. This is an additional reason for employing separate local observers. Finally, it is also to be noted that the results apply both in the case when normal controls are decentralized (functions of the local information only) as well as in the case when these controls are synthesized using additional system wide information.

APPLICATION TO INTERCONNECTED POWER SYSTEMS

Consider now the interconnected power system. The problem considered is that faced by one area in the interconnected system. This area is responsible only for alleviating its own power mismatch (and of course for helping out other areas if necessary). The problem then is how to determine whether a power mismatch has occurred within the area and if so to estimate the mismatch in a sufficiently short time period so that appropriate control action could be undertaken.

The problem is solved by taking advantage of the fact that tie-line power flows are measured in all control centers of an interconnected system. Omitting here the full model of the interconnected power system [10] and assuming for simplicity that the first area is connected only to area two at one point, we concentrate on the model of the first area. This model is given by

$$\begin{aligned} \dot{x}_1 &= A_1 x_1 + a_{12} p_{12} + g_1 z_1 \\ \dot{z}_1 &= 0 \end{aligned} \tag{19}$$

where $p_{12} = \Delta P_{12}/P^o_{12} = (P_{12} - P^o_{12})/P^o_{12}$ is the total tie-line power flow deviation. The problem is to estimate the value of unbalance z_1 (assuming all normal controls are incorporated in the plant matrix) given the measurements of the frequency $y = f_1 = [0 \ \ldots \ 1]x_1$ and tie line power flow, p_{12}. Thus, the full observer structure is

$$\dot{\bar{x}}_{1e} = \bar{A}_1 \bar{x}_{1e} + \bar{a}_{12} p_{12} + \bar{H}_1 (y_1 - y_{1e}) \tag{20}$$

where $\bar{A}_1$, $\bar{a}_{12}$, and $\bar{H}_1$ are extended plant matrix, tie-line input, and observer gains.

Clearly, as in the general case a reduced order observer could be used instead. In this case the observer structure from (17) is given by

$$\dot{\bar{x}}_{rl_e} = (\bar{A}_{rl} - \bar{H}_{rl}\bar{c}_{rl}) + \bar{a}_{12_r} p_{12} + \bar{H}_{rl} y_1 \tag{21}$$

with $\bar{A}_{rl}$, $\bar{c}_{rl}$, $\bar{a}_{12_r}$, and $\bar{H}_{rl}$ obtained from the spectral decomposition as in (16), taking into account the structure of c in the power system case. The structure of the reduced order observer is presented in Fig. 1.

The analysis of transient behavior of the reduced order observer parallels that conducted for general large scale systems with exactly the same conclusions. Thus, it can be used to estimate the power mismatch in an area utilizing only measurements of frequency and tie line power flows, in contrast to, e.g., [2]. It is also noted that the choice of observers based on the physically motivated models [5-7] is justified, the question being, should some additional slow dynamics also be extracted from the error to speed up the convergence. Conclusions presented in earlier sections indicate how such reduced order observers should be constructed.

In the power system problem the steady state value of the estimate is given by the steady state solution of (19) and (20) which produce

$$z^s_{1e} = \frac{c_1 A_1^{-1} g_1}{c_{r1} A_{r1}^{-1} g_{r1}} z_1 - \frac{c_{r1} A_{r1}^{-1} a_{12r} - c_1 A^{-1} a_{12}}{c_r A_{r1}^{-1} g_{r1}} p_{12}. \tag{22}$$

In order that p_{12} not affect the estimate it is necessary that $c_{r1} A_1^{-1} a_{12r} = c_1 A_1^{-1} a_{12}$. Due to the fact that in the power system model $g_1 = a_{12}$, it is sufficient to assure $c_{r1} A_{r1}^{-1} g_{r1} = c_1 A_1^{-1} g_1$ for zero steady state error. Reduced order observers obtained by spectral factorization achieve this automatically. The steady state estimate (22) can be reorganized and written in function of the available steady state values of the measurements, as

$$-z_{1e}^{s} = \frac{c_{r1}A_{r1}^{-1}a_{12r}}{c_{r1}A_{r1}^{-1}g_{r1}}\, p_{12} + \frac{1}{c_{r1}A_{r1}^{-1}g_{r1}}\, f_{1} . \tag{23}$$

In view of the fact that $c_{r1}A_{r1}^{-1}g_{r1} = c_{r1}A_{r1}^{-1}a_{12r}$, and that the frequency response characteristic of the system is defined by

$$B_{o} = \frac{1}{c_{r1}A_{r1}^{-1}g_{r1}} \text{ (Mw/Hz)} \tag{24}$$

it follows that

$$-z_{1e}^{s} = p_{12} + B_{o}f_{1} = ACE_{1} \tag{25}$$

which is precisely the area control error for the area considered with $B = B_o$, and B_o the area frequency response characteristic. Thus, it is clear that the introduced observer is completely compatible with the standardly used method of computing the area control error which is used as an input for automatic generation control. The distinction is that the method as developed here is also suitable for fast estimation of large power mismatch in short time intervals following the occurrence of the mismatch, whereas the area control concept was developed to remove power mismatch after the onset of steady-state conditions in the interconnected system.

CONCLUSION

The described procedure presents a methodological way of constructing detector-identifiers of power mismatch in interconnected systems utilizing only frequency and tie-line flow measurements. The main goal is to determine sufficiently accurate, and yet sufficiently fast detectors so that estimation is achieved in the available time interval following the outage so that the system model remains valid and that centralized structural control actions on demand could be undertaken before automatic and uncoordinated action is triggered by time-delayed settings of over-load protection. Preliminary simulation results were presented in [11]. In further development of the procedure the many practical problems related to the modeling issues, as well as to the adequacy of presently employed means of measuring tie-line power flows and frequency from the standpoint of fast detection will be considered. Modeling of power systems by a larger number of equivalent plants, and of including additional dynamic phenomena in the model of the system do not pose principal problems. It is also possible to identify multiple points of power mismatch monitoring in the area, which will give rise to the problem of observing a vector constant disturbance, which can be solved in full analogy with the described procedure, but will necessarily increase the observer dimension. Unresolved issues are related to the adequacy of linearized model for unbalances of order 5-10% of the system load, and to the needs of modifying the model if generation outages of the same order occur. These questions will be studied through system simulations with nonlinear models, and in testing the procedure in applications.

REFERENCES

[1] Zaborszky, J., K. Whang, and K. Prasad, "Monitoring, Evaluation and Control of Power System Emergences," Engineering Foundation Conference on Systems Engineering for Power - Emergency Operating State Control, Davos, Switzerland, Oct. 1-5, 1979.

[2] Glavitsch, H. and F. Galiana, "Load-Frequency Control with Particular Emphasis on Thermal Power Stations," in Real-Time Control of Electric Power Systems, E. Handschin, Ed., 1972.

[3] Cohn, N., Control of Generation and Power Flow on Interconnected Systems, J. Wiley, New York, 1966.

[4] Elgerd, O. L. and C. E. Fosha, Jr., "Optimum Megawatt-Frequency Control of Multiarea Electric Energy Systems," IEEE Trans. on PAS, Vol. PAS-89, No. 4, Parts I and II and discussion of papers, April 1970, pp. 556-577.

[5] Rabinovic, R. S., "Automatic Frequency Load Shedding in Power Systems," Energia, Moscow, 1980 (in Russian).

[6] Bongers, C., E. Grebe, E. Handschin, H. Haubrich, "Identification and Control of the Mid-Term Dynamics Behavior of Electric Power Systems," Power System Computing Conference, 7 PSCC, Lausane, July 1981, pp. 773-780.

[7] Welfonder, E., A. Lampert, F. Heilemann, M. Schmieg, "Applied Investigation of the Control Behavior of Different Power Systems by Simulation Studies and Measurements," Power System Computing Conference, 7 PSCC, Lausane, July 1981, pp. 781-794.

[8] Medanic, J., "Geometric Properties and Invariant Manifolds of the Riccati Equation," IEEE Trans. on Automatic Control, Vol. AC-27, No. 3, June 1982, pp. 670-677.

[9] Clements, D. J., B. D. O. Anderson, "Polynomial Factorization via the Riccati Equations," SIAM J. Appl. Math., Vol. 31, No. 1, July 1976.

[10] Calovic, M., "Dynamic State Space Models of Electric Power Systems," University of I-linois, Urbana, 1971.

[11] Cukalevski, N., M. Calovic, J. Medanic, "Low Order Observer a Detector-Identifier of Load Unbalances for Emergency Control," CIGRE/IFAC Symposium on Control Applications for Power System Security, Florence, Italy, Sept. 26-28, 1983.

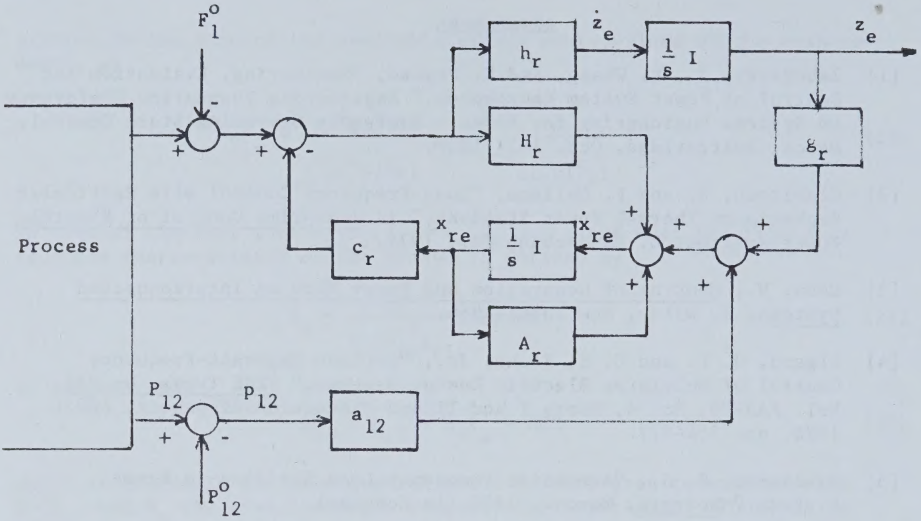

F_1^o
$\dot{z}_e$
z_e
h_r
$\frac{1}{s}I$
H_r
g_r
Process
c_r
x_{re}
$\frac{1}{s}I$
$\dot{x}_{re}$
A_r
P_{12}
P_{12}
a_{12}
P_{12}^o

Fig. 1. Reduced order observer for an interconnected power system.

APPENDIX

Partitioning (5) and defining x_1 and x_2 with model

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{A1}$$

$$y = [C_1 \quad C_2] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

apply a similarity transformation [8,9] to (A1) to obtain

$$\begin{aligned} T^{-1}\dot{\bar{x}} &= T^{-1}\bar{A}\,T\,T^{-1}\bar{x} \\ y &= \bar{C}\,T\,T^{-1}\bar{x} \end{aligned} \tag{A2}$$

where T has the form

$$T = \begin{bmatrix} I & -K \\ O & I \end{bmatrix} \tag{A3}$$

with K the solution of algebraic Riccati equation

$$R(K) = A_{12} + KA_{22} - A_{11}K - KA_{21}K = 0. \tag{A4}$$

Introducing $x' = T^{-1}\bar{x}$, or in matrix form

$$\begin{bmatrix} x_1' \\ x_2' \end{bmatrix} = \begin{bmatrix} I & K \\ 0 & I \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 + Kx_2 \\ x_2 \end{bmatrix}. \tag{A5}$$

The transformed state and output equation become

$$\begin{bmatrix} \dot{x}_1' \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} F_{11} & 0 \\ A_{21} & F_{22} \end{bmatrix} \begin{bmatrix} x_1' \\ x_2 \end{bmatrix} \tag{A6}$$

$$y = [C_1 \quad C_2] \begin{bmatrix} I & -K \\ 0 & I \end{bmatrix} \begin{bmatrix} x_1' \\ x_2 \end{bmatrix} = C_1x_1' + (C_2 - C_1K) \tag{A7}$$

where F_{11} and F_{22} are given by

$$\begin{aligned} F_{11} &= A_{11} + KA_{21} \\ F_{22} &= A_{22} - A_{21}K. \end{aligned} \tag{A8}$$

The transformation T, (A3), achieves a spectral decomposition of the system (A1), since the system is transformed into a series connection of two subsystems, (where S(X) denotes the spectrum of X) with

$$S(\bar{A}) = S(F_{11}) \cup S(F_{22}). \tag{A9}$$

To achieve the particular slow-fast decomposition, such that $S(F_{11})$ contains the fast eigenvalues and $S(F_{22})$ contains the slow eigenvalues it is necessary to select the solution of (A4) corresponding to this decomposition. The relationships of system decompositions to invariant subspaces of the system matrix and to construction of the solutions of (A4) corresponding to particular solutions are well understood [8,9], and the reader is referred to these references for further details.

STRUCTURALLY STABLE FEEDBACK CONTROL OF SINGULAR SYSTEMS

DANIEL COBB
Dept. of Electrical Engineering
University of Toronto
Toronto, Ontario M5S 1A4
Canada

ABSTRACT

Optimal regulation of linear systems of the form $E\dot{x} = Ax+Bu$, with E singular and quadratic cost, is considered. It is shown that the nonuniqueness of the optimizing feedback matrix can be exploited to give a closed loop system which remains stable in the presence of small perturbations (structurally stable). Although the problem of structural stability is not fully understood in general, a complete solution is given for an important special case.

INTRODUCTION

Systems of the form

$$E\dot{x} = Ax+Bu \tag{1}$$

have been treated in [1]-[5]. In [4] it is demonstrated that (1) may be decomposed into two subsystems

$$\dot{x}_s = A_s x_s + B_s u \tag{2a}$$

$$A_f \dot{x}_f = x_f + B_f u \tag{2b}$$

of dimension r and n-r respectively, where A_f is nilpotent and $x = \begin{bmatrix} x_s \\ x_f \end{bmatrix}$. It is shown in [5] that subsystem (2b) may have impulses present in the unforced solution. We have proven in [2] that all impulses in (2b) may be eliminated by applying a feedback matrix to the system if and only if

$$\mathrm{Im} A_f + \mathrm{Ker}\, A_f + \mathrm{Im}\, A_f = \mathbb{R}^{n-r} \tag{3}$$

We have also proven in [3] that an input u* exists which minimizes

$$J = \int_o^\infty ||x(t)||^2 + ||u(t)||^2 dt \tag{4}$$

with respect to (1) if and only if (2a) is stabilizable and (3) holds. In this case, u* is unique and can be implemented with a feedback matrix K which is not unique. (Impulses must necessarily be eliminated for optimality.) We are interested in distinguishing among the various choices of K.

The following example illustrates the problem:

$$\begin{bmatrix} o & 1 \\ o & o \end{bmatrix} \dot{x} = x + \begin{bmatrix} o \\ 1 \end{bmatrix} u \tag{5}$$

Such a system may arise from a singular perturbation problem involving

$$\begin{bmatrix} -\varepsilon & 1 \\ o & -\varepsilon \end{bmatrix} \dot{x} = x + \begin{bmatrix} o \\ 1 \end{bmatrix} u \tag{6}$$

where $\varepsilon > o$. For $u = o$ and $x(o) = \begin{bmatrix} o \\ 1 \end{bmatrix}$, (6) has the solution

$$x_\varepsilon(t) = \begin{bmatrix} -\frac{t}{\varepsilon^2} e^{-\frac{t}{\varepsilon}} \\ e^{-\frac{t}{\varepsilon}} \end{bmatrix} \tag{7}$$

As $\varepsilon \to o^+$, $x_\varepsilon \to \begin{bmatrix}-\delta \\ o\end{bmatrix}$. The limit of x_ε can be shown to be the solution of (5) for the same u_o and $x(o)$. Following [3], optimality is achieved for all feedback matrices of the form

$$K = [\alpha \quad \sqrt{2}\alpha - 1] \tag{8}$$

where $\alpha \neq o$. If we choose $\alpha > o$ the solution of the closed-loop system converges as $\varepsilon \to o^+$. However, for $\alpha < o$, the closed loop system has one eigenvalue tending to $+\infty$. Clearly, this is an undesirable situation. We would like to identify, in the general case, the class of feedback matrices which are not only optimal at $\varepsilon = o$, but which also yield convergence of solutions as $\varepsilon \to o^+$.

PROBLEM FORMULATION

Actually, we will consider a more general type of perturbation than the one described above. Our main assumption will be that, whatever perturbation of (1) exists, the corresponding solutions of (1) converge as $\varepsilon \to o^+$ for every possible value of $x(o)$. This, after all, is saying nothing more than that (1) is a "good" idealization of the physical system being modelled.

Definition: We say that a feedback matrix K yields a structurally stable closed-loop system if the closed-loop solutions converge under all perturbations which guarantee convergence of solutions of (1).

Symbolically, the situation can be described as follows: Applying feedback K to (1) yields a system which can be decomposed as in (2) to give

$$\dot{x}_{sK} = A_{sK}x_{sK}+B_{sK}u \tag{9a}$$

$$A_{fK}\,\dot{x}_{fK} = x_{fK}+B_{fK}u \tag{9b}$$

The closed-loop system is structurally stable if and only if $e^{tA_{fK}^{-1}}$ is convergent whenver $e^{tA_f^{-1}}$ converges. (Here we are only considering perturbations which make E nonsingular. The general case can also be handled in this framework, but with increased notational complexity.)

The central problem is that of determining which values of K from the optimal class yield a structurally stable system. Structural instability is clearly unacceptable since the idealized closed-loop model would not accurately predict the behaviour of the physical system in question.

THE CASE OF CYCLIC A_f

When A_f in (2b) is cyclic at $\varepsilon = o$, a solution is readily obtained. We are able to prove the following series of results leading up to an algorithm for choosing K:

1) There exists a subspace S^* with $S^* \oplus \operatorname{Ker} E = \mathbb{R}^n$ such that the class of optimizing feedback matrices is the linear variety in $\mathbb{R}^{m \times n}$ formed by adding to any optimal K all matrices $\bar{K}$ satisfying

$$\operatorname{Ker} \bar{K} \supset S^* \tag{10}$$

2) As a result of 1), if the class of optimal feedback matrices is nonempty, it contains at least one member that yields structural stability.

3) If coordinates are chosen in (2b) so that A_f is in Jordan form at $\varepsilon = o$, a matrix K_f which is optimal for (2b) alone yields structural stability if and only if

$$b_{n-r} k_1 > o \quad \text{for } n\text{-}r \text{ odd} \tag{11a}$$

$$b_{n-r} k_1 < o \quad \text{for } n\text{-}r \text{ even} \tag{11b}$$

where $B_f = \begin{bmatrix} b_1 \\ \vdots \\ b_{n-r} \end{bmatrix}$, $K_f = [k_1 \cdots k_{n-r}]$.

4) If ε-dependent matrices E, A_1, and A_2 are given with $A_1(o) = A_2(o)$ and $E(o)\dot{x} = A_1(o)x$ a closed-loop optimal system, then the solutions of $E\dot{x} = A_1 x$ converge if and only if those of $E\dot{x} = A_2 x$ do also.

Utilizing 1) - 4), we have obtained an algorithm for finding an optimal feedback matrix which yields structural stability once any optimal matrix is known:

a) Starting with (1), change coordinates to decouple the system as in (2) with A_f in Jordan form at $\varepsilon = o$.

b) Alter the (r+1)th column of the given optimal K to satisfy (11) by adding an appropriate $\bar{K}$ satisfying (10). The altered K is also optimal and is guaranteed to yield structural stability.

c) Transform back to the original coordinates.

REFERENCES

[1] S.L. Campbell, Singular Systems of Differential Equations, Pitman, 1980.

[2] D. Cobb, "Feedback and Pole Placement in Descriptor Variable Systems," Int. J. Control, Vol. 33, No. 6, 1981, 1135-1146.

[3] D. Cobb, "Descriptor Variable Systems and Optimal State Regulation," IEEE Trans. Auto. Control, Vol. 28, No. 5, 1983, 601-611.

[4] F.R. Gantmacher, The Theory of Matrices, Chelsea, 1960.

[5] G.C. Verghese, "A Generalized State-Space for Singular Systems," IEEE Trans. Auto. Control, Vol. 26, No. 4, 1981, 811-831.

SPECTRAL ASSIGNMENT USING REDUCED-ORDER MODELS

PRESTON I. CARRAWAY AND ROLAND R. MIELKE
Department of Electrical Engineering
Old Dominion University
Norfolk, Virginia

ABSTRACT

The feasibility of using reduced-order models with eigenvalue/eigenvector assignment procedures is investigated. First, a reduced-order model which retains essential system characteristics is formulated. Then a constant state feedback matrix which assigns desired closed-loop eigenvalues and approximates specified closed-loop eigenvectors is calculated for the reduced-order model. It is shown that the eigenvalue and eigenvector assignments made in the reduced-order system are retained when the feedback matrix is implemented about the full-order system. In addition, those modes which are not included in the reduced-order model remain unchanged in the closed-loop full-order system.

INTRODUCTION

During the past fifteen years significant advances have been made toward developing viable synthesis techniques for multivariable feedback control systems. Notable among these techniques is the eigenvalue/eigenvector assignment procedure. Early studies in this area focused on an algorithmic formulation of the spectral assignment problem by Srinathkumar [1], while later studies included a geometric formulation of the same problem by Moore [2], Kimura [3], and Davison and Wang [4]. Based on these theories, the authors and colleagues have developed design procedures for approximating desired mode mixing [5], reducing eigensystem sensitivity to variations in plant parameters [6], reducing the effects of actuator noise on system performance [7], and modifying the resultant feedback gain matrix to satisfy specified gain constraints. Recently these procedures have been incorporporated in a spectral assignment computer aided design package [9]. A deficiency in all work concerning eigenvalue/eigenvector assignment procedures is an absence of application of these techniques to real world design problems. A primary factor contributing to this problem is the lack of understanding of how to use reduced-order models with spectral assignment procedures.

Models representing the behavior of physical systems often consist of a very large number of coupled, linear differential equations. Such models are difficult to use when designing control systems due to excessive requirements for computer time and memory, and to the numerical analysis problems inherently present when dealing with large systems of equations. It is therefore desirable to develop a design procedure which utilizes reduced order system models. Simplification of large order dynamic systems has received the attention of many researchers in recent years. The major difficulty with this work is that only open-loop system behavior is approximated. Of concern when using reduced-order models with eigenvalue/eigenvector assignment procedures is the fact that while the reduced-order model may approximate open-loop system behavior, the modeling error may be so great or of such a nature that actual closed-loop system performance is not acceptable. Also of concern is the closed-loop behavior of those modes of the original system which are not included in the reduced-order model. The

---

This work was supported by the NASA Langley Research Center under Grant NSG-1650.

purpose of this paper is to investigate the feasibility of using reduced-order models with eigenvalue/eigenvector assignment procedures.

The design procedure consists of three steps. First, a reduced-order model which retains essential system characteristics is formulated. The reduced-order model includes all eigenvalues which are to be reassigned via an eigenvalue/eigenvector assignment procedure. However, it is not necessary to retain all dominant system modes in this model. Those modes not included in the reduced-order model will remain unchanged in the final closed-loop system. Second, the eigenvalue/eigenvector assignment procedures are used to calculate a constant state feedback matrix for the reduced-order model. This matrix is chosen to assign specified eigenvalues and approximate desired eigenvectors in the reduced order system. Third, the state feedback matrix computed for the reduced-order system is implemented about the full-order system. It is shown that the eigenvalue and eigenvector assignments made in the reduced-order system are retained in the closed-loop full order system. In addition, those modes not included in the reduced-order model remain unchanged in the closed-loop full order system.

DESIGN PROCEDURE

The first step of the design procedure is to formulate a reduced order model that contains the eigenvalues of the full order original system that are to be reassigned. Let the original system model be described by the state equation

$$\dot{x} = Ax + Bu \tag{1}$$

where $x \varepsilon R^n$ and $u \varepsilon R^m$. It is assumed that the eigenvalues of A are distinct and denoted by $\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$. The corresponding modal matrix for A is denoted by $V = [v_1, v_2, \cdots, v_n]$. The A matrix is diagonalized by defining a new state variable with the transformation

$$z = V^{-1}x \; . \tag{2}$$

The resulting state equation is

$$\dot{z} = \Lambda z + \hat{B}u \tag{3}$$

where $\Lambda = V^{-1}AV = \text{diagonal } \{\lambda_1, \lambda_2 \cdots \lambda_n\}$, and $\hat{B} = V^{-1}B$. The system is now partitioned in order to separate the eigenvalues to be retained in the reduced-order model from those that will remain unchanged in the full order system model.

$$\begin{bmatrix} \dot{z}_1 \\ \hline \dot{z}_2 \end{bmatrix} = \left[\begin{array}{c|c} \Lambda_1 & 0 \\ \hline 0 & \Lambda_2 \end{array}\right] \begin{bmatrix} z_1 \\ \hline z_2 \end{bmatrix} + \begin{bmatrix} \hat{B}_1 \\ \hline \hat{B}_2 \end{bmatrix} u \tag{4}$$

where $z_1 \varepsilon R^p$ and $z_2 \varepsilon R^{n-p}$. The eigenvalues to be retained in the reduced order model must be contained in Λ_1 and those which are not retained must be contained in Λ_2. The reduced order model is thus described by the state equation

$$\dot{z}_1 = \Lambda_1 z_1 + \hat{B}_1 u \; . \tag{5}$$

The second step of the design procedure is to calculate a constant state feedback matrix for this reduced order model using eigenvalue/eigenvector assignment procedures. These procedures will assign specified eigenvalues and approximate desired eigenvectors in the closed-loop system. The eigenvalues and eigenvectors that should be assigned in the reduced-order model to achieve specified eigenvalues and partial eigenvector assignment in the full order system are determined by completing the third step of the design procedure. The feedback matrix computed for the reduced order model is implemented about the full order system and the relationships between eigenvalues and eigenvectors of the reduced order model and those in the full order system model are investigated.

The relationship between eigenvalues of the reduced order and full order system models is determined first. Given a state feedback matrix, $\hat{F}$, for the reduced order system and the control law

$$u = \hat{F}z_1 \ , \qquad (6)$$

the closed loop reduced-order system model is written as

$$\dot{z}_1 = (\Lambda_1 + \hat{B}_1\hat{F})z_1 \ . \qquad (7)$$

Equation (6) is rewritten as

$$u = [\hat{F} \vdots 0] \begin{matrix} z_1 \\ z_2 \end{matrix} = \tilde{F}z \ . \qquad (8)$$

Substituting for z from equation (2) gives

$$u = \tilde{F}V^{-1}x = \bar{F}x \ . \qquad (9)$$

Hence the closed loop full order system is written as

$$\begin{aligned} \dot{x} &= Ax + B\bar{F}x \\ &= [A + B\bar{F}]x \ . \end{aligned} \qquad (10)$$

The eigenvalues of the full order system are therefore the eigenvalues of $[A + B\bar{F}]$. Substituting from equations (2), (3), and (9) into equation (10) results in the equation

$$[A + B\bar{F}] = V[\Lambda + \hat{B}\tilde{F}]V^{-1} \ . \qquad (11)$$

Since $[A+B\bar{F}]$ and $[\Lambda+\hat{B}\tilde{F}]$ are related by a similarity transformation they have the same eigenvalues. The matrix $[\Lambda + \hat{B}\tilde{F}]$ is expanded as

$$[\Lambda+\hat{B}\tilde{F}] = \left[\begin{array}{c|c} \Lambda_1+\hat{B}_1\hat{F} & 0 \\ \hline \hat{B}_2\hat{F} & \Lambda_2 \end{array}\right] \ . \qquad (12)$$

The eigenvalues of this matrix are obviously the eigenvalues of $[\Lambda_1 + \hat{B}_1\hat{F}]$ and $[\Lambda_2]$. Thus the designer is able to reassign the p eigenvalues included in the reduced order model without modifying the (n-p) original system eigenvalues which were not included in the reduced order model.

The relationship between eigenvectors of the reduced order and full order system models is determined next. The eigenvector equation for $[\Lambda + \hat{B}\tilde{F}]$ is written as

$$[\lambda_i-\Lambda-\hat{B}\tilde{F}]\hat{V}_i = \left[\begin{array}{c|c} \lambda_i I-\Lambda_1-\hat{B}_1\hat{F} & 0 \\ \hline -\hat{B}_2\hat{F} & \lambda_i I-\Lambda_2 \end{array}\right] \left[\begin{array}{c} \hat{V}_{1i} \\ \hline \hat{V}_{2i} \end{array}\right] = 0 \qquad (13)$$

which yields the two equations

$$[\lambda_i I-\Lambda_1 - \hat{B}_1\hat{F}]\hat{V}_{1i} = 0 \qquad (14)$$

$$[-\hat{B}_2\hat{F}]\hat{V}_{1i} + [\lambda_i I-\Lambda_2]\hat{V}_{2i} = 0 \qquad (15)$$

If λ_i is an eigenvalue of $[\Lambda_1 + \hat{B}_1\hat{F}]$, but not an eigenvalue of $[\Lambda_2]$, then from equation (14) it follows that $\hat{V}_{1i}$ is an eigenvector of $[\Lambda_1+\hat{B}_1\hat{F}]$. Since λ_i is not an eigenvalue of $[\Lambda_2]$, $[\lambda_i I-\Lambda_2]$ is nonsingular. Thus, from equation (15)

$$\hat{V}_{2i} = [\lambda_i I-\Lambda_2]^{-1}\hat{B}_2\hat{F}\hat{V}_{1i} \ . \tag{16}$$

Therefore $\hat{V}_i$ is written

$$\hat{V}_i = \left[\begin{array}{c} I \\ \hdashline [\lambda_i I-\Lambda_2]^{-1}\hat{B}_2\hat{F} \end{array}\right] \hat{V}_{1i} \tag{17}$$

Equation (17) illustrates that the first p elements of $\hat{V}_i$ can be assigned in the reduced-order model while the remaining n-p elements are linear combinations of $\hat{V}_{1i}$.

On the other hand, if λ_i is an eigenvalue of Λ_2 and not an eigenvalue of $[\Lambda_1 + \hat{B}_1\hat{F}]$ then $[\lambda_i I-\Lambda_1-\hat{B}_1 F]$ is nonsingular. Therefore, for equation (14) to be satisfied it follows that

$$\hat{V}_{1i} = 0 \ . \tag{18}$$

From equation (15) it follows that $\hat{V}_{2i}$ is an eigenvector of Λ_2 for eigenvalue λ_i. Since Λ_2 is a diagonal matrix, $\hat{V}_{2i}$ is written as

$$\hat{V}_{2i} = [0,\cdots,0,K_i,0,\cdots,0]^T \tag{19}$$

where K_i is a nonzero constant. Therefore $\hat{V}_i$ is written as

$$\hat{V}_i = \left[\begin{array}{c} 0 \\ \hdashline \hat{V}_{2i} \end{array}\right] \tag{20}$$

Let $\overline{V}_i$ be the eigenvector of $[A + B\overline{F}]$ corresponding to eigenvalue λ_i. From equation (11) it is known that for eigenvalues of $[\Lambda_1 + \hat{B}\hat{F}]$, the eigenvectors $\overline{V}_i$ and $\hat{V}_i$ are related by the transformation

$$\overline{V}_i = V\hat{V}_i \ . \tag{21}$$

Expanding equation (21) then gives

$$\left[\begin{array}{c} \overline{V}_{1i} \\ \hdashline \overline{V}_{2i} \end{array}\right] = \left[\begin{array}{c} V_{11}+V_{12}[\lambda_i I-\Lambda_2]^{-1}\hat{B}_2\hat{F} \\ \hdashline V_{21}+V_{22}[\lambda_i I-\Lambda_2]^{-1}\hat{B}_2\hat{F} \end{array}\right] \hat{V}_{1i} \ . \tag{22}$$

In many cases, such as when the reduced order model contains only the dominant modes of the system, $V_{11} \doteq V_{11} + V_{12}[\lambda_i I-\Lambda_2]^{-1}\hat{B}_2\hat{F}$. Then the top p components of the first p eigenvectors are assigned by choosing $\hat{V}_{1i}$ as

$$\hat{V}_{1i} = V_{11}^{-1}\overline{V}_{1i} \tag{23}$$

When the above approximation does not apply, an initial assignment of $\hat{V}_{1i}$ is made using equation (23), $\hat{F}$ is calculated, and the error between the desired top p components of the first p eigenvectors and the actual assignment is calculated. A gradient search procedure is then used to reduce this error.

From equation (11) it is known that for λ_i which are eigenvalues of $[\Lambda_2]$

$$\bar{V}_i = V\hat{V}_i = \left[\begin{array}{c} V_{12} \\ \hline V_{22} \end{array}\right]\hat{V}_{2i} \; . \qquad (24)$$

Substituting for $\hat{V}_{2i}$ from equation (19) gives

$$V_i = K_i \left[\begin{array}{c} V_{12} \\ \hline V_{22} \end{array}\right] . \qquad (25)$$

Therefore, the last n-p eigenvectors in the closed loop full order system are the original open-loop eigenvectors. Thus eigenvectors corresponding to the eigenvalues retained in the full order system model remain the same in the final closed loop system.

DESIGN EXAMPLE

An example is given to illustrate the design procedure. The lateral axis model of an L-1011 aircraft is used as the original full order system model. The state vector x is given by

$x_1 = r$ = yaw rate (rad/sec)

$x_2 = \beta$ = sideslip angle (rad)

$x_3 = p$ = roll rate (rad/sec)

$x_4 = \phi$ = bank angle (rad)

$x_5 = \delta_r$ = rudder deflection (rad)

$x_6 = \delta_a$ = aileron deflection (rad)

The A and B system matrices are given by

$$A = \begin{bmatrix} -0.154 & 1.54 & -0.0042 & 0 & -0.744 & -0.032 \\ -0.996 & -0.117 & -0.000295 & 0.0386 & 0.02 & 0 \\ 0.249 & -5.2 & -1.0 & 0 & 0.337 & -1.12 \\ 0 & 0 & 1.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -20.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -25.0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 20 & 0 \\ 0 & 25 \end{bmatrix}$$

The system input, u, consists of components

$$u_1 = \delta r_c = \text{rudder command (rad)}$$

$$u_2 = \delta a_c = \text{aileron command (rad)} .$$

The open loop eigenvalues of this system are

$$\begin{aligned} \lambda_{1,2} &= -0.08819 \pm j1.269 && \text{- dutch roll} \\ \lambda_3 &= -1.085 && \text{- roll subsistance mode} \\ \lambda_4 &= -0.009165 && \text{- spiral mode} \\ \lambda_5 &= -20.0 && \text{- rudder mode} \\ \lambda_6 &= -25.0 && \text{- aileron mode} \end{aligned}$$

It is desirable to achieve the following closed loop eigenvalues:

$$\lambda_{1,2} = -1.5 \pm j1.5 \text{ - dutch roll}$$

$$\lambda_{3,4} = -2.0 \pm j1.5 \text{ - roll mode}$$

It is also desirable to decouple the roll mode and dutch roll mode. This is achieved by the following desirable eigenvector assignment corresponding to eigenvalues λ_1 through λ_4 ,

$$v_1 = \begin{bmatrix} 1 \\ x \\ 0 \\ 0 \\ x \\ x \end{bmatrix}, \; v_2 = \begin{bmatrix} x \\ 1 \\ 0 \\ 0 \\ x \\ x \end{bmatrix}, \; v_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ x \\ x \\ x \end{bmatrix}, \; v_4 = \begin{bmatrix} 0 \\ 0 \\ x \\ 1 \\ x \\ x \end{bmatrix}$$

where x denotes "don't care".

Since λ_5 and λ_6 are not specified, and the rudder and aileron deflections are also unspecified, no reassignments of these modes will be made. Therefore they are not included in the reduced order model.

The reduced order model obtained from equations (2), (3), (4), and (5) has the following system matrices.

$$\Lambda_1 = \begin{bmatrix} -0.08819 & 1.269 & 0 & 0 \\ -1.269 & -0.08819 & 0 & 0 \\ 0 & 0 & -1.085 & 0 \\ 0 & 0 & 0 & -0.009165 \end{bmatrix} \quad \hat{B}_1 = \begin{bmatrix} -1.706 & -0.02580 \\ 0.3961 & 0.06988 \\ -0.2772 & -0.2878 \\ -0.2698 & -0.1528 \end{bmatrix}$$

A partial eigenvector assignment which achieves the desired decoupling of the roll mode and dutch roll mode is

$$V_{1,2} = \begin{bmatrix} 1 \pm j1 \\ 1 \pm j1 \\ 0 \\ 0 \end{bmatrix} \qquad V_{3,4} = \begin{bmatrix} 0 \\ 0 \\ 1 \pm j1 \\ +1 \pm j1 \end{bmatrix} .$$

An initial approximation of this desired eigenvector assignment in the reduced order model is made using equation (23). The desired eigenvectors in the reduced-order model are

$$\hat{V}_{1,2_{DES}} = \begin{bmatrix} 2.875 \pm j2.875 \\ 2.273 \pm j2.273 \\ 0.9696 \pm j0.9696 \\ 0.3883 \pm j0.3883 \end{bmatrix} \qquad \hat{V}_{3,4_{DES}} = \begin{bmatrix} -0.1332 & \mp j0.1332 \\ -0.03272 & \mp j0.03272 \\ 0.2146 & \pm j0.2146 \\ 0.2494 & \pm j0.2494 \end{bmatrix}$$

The desired eigenvalues are assigned and $\hat{V}_1$ through $\hat{V}_4$ are approximated in the reduced-order model using a spectral assignment computer aided design package [9]. The following eigenvectors are the approximations achieved in the reduced-order model.

$$\hat{V}_{1,2_{ACT}} = \begin{bmatrix} 3.593 & \pm j2.048 \\ -0.08381 & \pm j1.872 \\ 0.6264 & \pm j0.7129 \\ 0.5439 & \pm j0.3148 \end{bmatrix} \qquad \hat{V}_{3,4_{ACT}} = \begin{bmatrix} -0.1059 & \mp j0.1131 \\ -0.04074 & \mp j0.08046 \\ 0.2358 & \pm j0.2914 \\ 0.1136 & \pm 0.05674 \end{bmatrix}$$

The partial assignment in the full-order system is found to be

$$V_1 = \begin{bmatrix} 1 \\ 0.304 + j0.344 \\ -0.556 + j0.143 \\ 0.233 + j0.138 \end{bmatrix} \qquad V_2 = \begin{bmatrix} 1.44 + j1.63 \\ 1 \\ -0.568 - j1.11 \\ 0.560 + j0.181 \end{bmatrix}$$

$$V_3 = \begin{bmatrix} -0.0351 - j0.0046 \\ -0.0069 - j0.0041 \\ 1 \\ -0.320 - j0.240 \end{bmatrix} \qquad V_4 = \begin{bmatrix} 0.077 + j0.045 \\ 0.020 + j0.002 \\ -2.0 - j1.5 \\ 1 \end{bmatrix}$$

A gradient search routine for the reduced-order model spectral assignment has been added as a new mode to the spectral assignment computer aided design package. It is now used to improve the initial eigenvector assignment while retaining the assigned eigenvalues. The final eigenvector assignment in the reduced-order model is

$$\hat{V}_{1,2} = \begin{bmatrix} 3.138 & \pm\, j1.788 \\ -0.1178 & \pm\, j1.649 \\ 0.8379 & \pm\, j0.6958 \\ 0.5847 & \pm\, j0.2436 \end{bmatrix} \qquad \hat{V}_{3,4} = \begin{bmatrix} -0.03131 & \mp\, j0.1079 \\ -0.1471 & \mp\, j0.03640 \\ 1.030 & \pm\, j0.4852 \\ 0.4137 & \pm\, j0.01440 \end{bmatrix}$$

which corresponds to the partial assignment in the full-order system of

$$V_1 = \begin{bmatrix} 1 \\ 0.3061 + j0.3510 \\ 0.2083 - j0.0156 \\ -0.0746 - j0.0642 \end{bmatrix} \qquad V_2 = \begin{bmatrix} 1.411 + j1.618 \\ 1 \\ 0.2687 + j0.3591 \\ -0.2093 - j0.0302 \end{bmatrix}$$

$$V_3 = \begin{bmatrix} -0.00321 - j0.0173 \\ -0.00571 + j0.0001 \\ 1 \\ -.3200 - 0.2400 \end{bmatrix} \qquad V_4 = \begin{bmatrix} 0.0323 - j0.0297 \\ -0.0116 - j0.0083 \\ -2.000 - j1.5000 \\ 1 \end{bmatrix}$$

The required gain matrix in the reduced order model is given by

$$\hat{F} = \begin{bmatrix} 1.494 & -1.449 & 0.1385 & -1.550 \\ -4.364 & 1.035 & -9.813 & 39.25 \end{bmatrix}$$

Using equations (8) and (9) the full-order system gain matrix is given by

$$\overline{F} = \begin{bmatrix} 3.669 & -3.137 & -0.1607 & -0.3476 & -0.1371 & 0.0023 \\ 1.342 & -5.807 & 2.749 & 5.481 & -0.0007 & -0.1194 \end{bmatrix}$$

CONCLUSION

A reduced order model was formulated containing eigenvalues of interest from an original full-order system. A constant state feedback matrix was designed for the reduced-order model that, when implemented about the full order system, reassigned the eigenvalues contained in the reduced-order model while those eigenvalues not included in the reduced-order system are retained. A partial eigenvector assignment was achieved for the eigenvectors corresponding to eigenvalues contained in the reduced-order model while the eigenvectors corresponding to eigenvalues not included in the reduced-order model are retained in the full-order system.

REFERENCES

[1] S. Srinathkumar, "Eigenvalue/Eigenvector Assignment Using Output Feedback," IEEE Trans. Automatic Control, Vol. AC-23, February 1978.

[2] B. C. Moore, "On the Flexibility Offered by State Feedback in Multivariable Systems Beyond Closed Loop Eigenvalue Assignment," IEEE Trans. Automatic Control, Vol. AC-21, October 1976.

[3] H. Kimura, "A Further Result on the Problem of Pole Assignment by Output Feedback," IEEE Trans. Automatic Control, Vol. AC-22, June 1977.

[4] E. J. Davison and S. H. Wang, "On Pole Assignment in Linear Multivariable Systems," IEEE Trans. Automatic Control, Vol. AC-20, August 1975.

[5] R. A. Maynard, R. R. Mielke, S. R. Liberty and S. Srinathkumar, "A New Spectral Synthesis Procedure for Multivariable Regulators," Allerton Conference on Communication, Control and Computing, Vol. 16, October 1978.

[6] R. R. Mielke, R. A. Maynard, S. R. Liberty and S. Srinathkumar, "Design of Robust State Feedback Controllers Via Eigenvalue/Eigenvector Assignment," Asilomar Conference on Circuits, Systems and Computers, Vol. 12, November 1978.

[7] R. A. Maynard, R. R. Mielke and S. R. Liberty, "Eigenvector Assignment for Noise Suppression," Southeastern Symposium on System Theory, Vol. 13, March 1981.

[8] T. M. Owens and R. R. Mielke, "A Spectral Assignment Gain Modification Procedure," Southeastern Symposium on System Theory, April 1982.

[9] R. R. Mielke, L. J. Tung and M. Marefat, "Design of Multivariable Feedback Controllers Via Spectral Assignment," Progress Report, NASA Grant NSG-1650, Old Dominion University Research Foundation, June 1982.

AN ITERATIVE ALGORITHM FOR OUTPUT FEEDBACK POLE PLACEMENT

S.P. BINGULAC
University of Belgrade,
Department of Organizational Sciences,
Beograd, Jove Ilica 154, Yugoslavia

V. MITROVIĆ
University of Belgrade,
Mechanical Engineering
Department, Yugoslavia

ABSTRACT

The paper describes an iterative algorithm for output feedback pole placement in linear multivariable dynamic systems. Compared to other existing pole placement algorithms, the algorithm does not require the usual condition $n < m+k$, n, m and k being dimensions of state, input and output vectors, respectively, which makes it more flexible and more applicable. In addition, if due to system structural properties, the exact solution does not exist, the algorithm renders an approximate solution which is, in most practical situtations, also applicable.

INTRODUCTION

In the last decade the problem of output feedback pole placement received a great attention in the control literature. Nevertheless, sufficient and necessary conditions for solving the pole placement problem by a proportional regulator has not yet been fully formulated. Consequently, neitheris the minimal order of the necessary dynamic compensator known. Some results in treating these problem are given in /1/, /2/ and /3/, but they are rather conservative which is particulary evident in the cases of higher order systems. Various pole placement algorithms have been suggested by different authors, such as /4/, /5/, /6/, etc. However, their advantages in terms of required conditions, convergence and applicability may be questioned. Hence, it would be desirable to have an algorithm which would

- require only the necessary condition /7/, i/e./. $m \cdot k = n$,
- converge to the solution, if it exists, or
- determine an approximate solution (with respect to a given criterion) if the exact solution does not exist, and
- yield the full rank regulator matrix.

This paper presents an iterative algorithm for solving the output feedback pole placement which has the above mentioned properites.

PROBLEM FORMULATION

Consider a realization (C,A,B) describing a linear system where dimension of output, state and input vectors are k, n and m, respectively.

Let $\Lambda_n = \{\lambda_i\}_{i=1}^{n}$, $\Lambda_n \subset C$, be an arbitrary symmetric set of complex numbers. Output feedback problem may be formulated as calculating (m x k) matrix K (if it exists) which satisfies

$$\sigma(A + B\,K\,C) = \Lambda_n \tag{1}$$

where $\sigma(X)$ denotes spectrum of matrix X. The same problem may also be formulated by

$$f(s;\ A+BKC) = f_d(s), \quad \forall s \in C \tag{2}$$

where $f(s;X) = \det\,(sI_n - X) = \sum_{i=0}^{n} f_{i+1} \cdot s^i \quad ; \quad f_{n+1} = 1$

is characteristic polynomial of matrix X, while f_d

$$f_d(s) = \sum_{i=0}^{n} f_{d_{i+1}} \cdot s^i , \; f_{d_{n+1}} = 1 \quad \text{is a monic polynomial}$$

associated with the a given set Λ_n. Obviously, eq. (2) will be satisfied if the corresponding coefficients of the polynomials $f(s;A+BKC)$ and $f_d(s)$ are equal, i.e. if:

$$f_d = f(A+BKC) = f(K) \tag{3}$$

where

$$f_d = \left[f_{d_1} , \ldots , f_{d_n}\right]^T \quad ; \quad f(K) = \left[f_1(K) , \ldots , f_n(K)\right]^T$$

Therefore, by eq. (3) the output feedback pole placement problem is reduced to the solution of system of n nonlinear equation with $m \cdot k$ unknowns, i.e. the elements of the matrix $K = (k_{ij})_m^k$.

SOLUTION OF THE PROBLEM

Applying the Taylor expansion, vector function $f(K)$ may be expressed as

$$f(K + \Delta K) = f(K) + (\partial f / \partial K)_K \, \Delta k + O(\Delta K) \tag{4}$$

In (4), $(\partial f / \partial K)_K$ denotes matrix of partial derivatives of $f_i(K)$ with respect to elements k_{ij}.

Note that $(m \times k)$ matrix ΔK and $m \cdot k$ dimensional vector Δk are related to each other by:

$$\Delta K = \left[k^1 \; \ldots\ldots \; k^k\right] \quad (m \times k)$$

$$\Delta k = \begin{bmatrix} k^1 \\ \vdots \\ \vdots \\ k^k \end{bmatrix} \begin{matrix} (m \times 1) \\ \\ (mk \times 1) \\ \\ (m \times 1) \end{matrix} \tag{5}$$

Assuming that a triple (C,A,B) and the vector f_d of compatible dimensions are given, then, based on (4), the following iterative algorithm for solving the output feedback pole placement problem may be formulated.

Pole placement (PP) algorithm

Step 1. Set $i:=0$, $K_i:=K_{in}$, $i_m:=i_{max}$, $e:=e_0$, where K_{in}, i_{max} and e_0 are initial guess for the matrix K, maximum number of iterations and a sufficiently small positive number, respectively.

Step 2. Evaluate $f(K_i)$ and $(\partial f / \partial K)_{K_i}$. An efficient method for calculating $(\partial f / \partial K)$ and $f(K)$ is given in Appendix A.

Step 3. If $|f_{d_j} - f_j(K_i)| < e \, |f_{d_j}|$, $j \in [1,n]$, set $K:=K_i$ and stop; else, go to step 4.

Step 4. Solve $(\partial f / \partial K)_{K_i} \cdot \Delta k_i = f_d - f(K_i)$. Cases occuring in solving a system of linear algebraic equations are discussed in Appendix C.

Step 5. Rearrange elements of Δk_i into ΔK_i accoring to (5) and set $K_{i+1}:=K_i + \Delta K_i$ and $i:=i+1$.

Step 6. If $i < i_m$, go to step 2; else, stop.

It should be noted that the algorithm PP does not guarantee convergence of the iterative process. Moreover, in most practical cases the algorithm would not converge. In order to guarantee the convergence to a solution, the step 5 of the Algorithm PP should be replaced by the following stabilizing procedure.

Stabilizing procedure (SP)

```
Step 1. Set $J:=0$, $d_1:= \| f_d - f(K_i) \|$, $b:=0.5$, $a_j:=2$.
Step 2. Set $a_{j+1}:=a_j \cdot b$, $j:=j+1$.
Step 3. If $j > i_m$ stop, else, go to step 4.
Step 4. Set $K^*:=K_i + a_j \cdot \Delta K_i$ and evaluate $F(K^*)$, (see, Appendix B).
Step 5. Set $d_2:= \| f_d - f(K^*) \|$.
Step 6. If $d_2 > d_1$ go to step 2; else, go to step 7.
Step 7. Set $K_i:=K^*$ and return to the step 6 of the algorithm PP.
```

Obviously, the suggested procedure SP prevents the divergence of the algorithm PP. In other words, the algorithm will either converge to a solution or oscilate in the vicinity of a local minimum. In the latter case another choice of an initial guess K_{in} would be appropriate. The advantages of the algorithm are as follows.

(1). If the algorithm does not converge to a solution, it will render an approximate solution K satisfying a minumum of $\| f_d - f(K) \|$, i.e.

$$\| f_d - f(\tilde{K}) \| = \min_K \| f_d - f(K) \|$$

which is in most cases satisfactory.

(2). The algorithm does not require the usual condition $m + k > n$, to restrictive for high order systems. In many cases the necessary conditon for a generic solution /7/, i.e. $m \cdot k = n$, also becomes sufficient. Even for $m \cdot k < n$, the algorithm gives an approximate solution.

(3). The algorithm may equaly be applied to nonminimal triples (C,A,B). In this case, however, the set Λ_n should contain all uncontrollable and observable modes.

(4). Normally, the algorithm gives full rank matrix K.

COMPUTATIONAL EXAMPLE

In order to illustrate applicability of the suggested algorithm, consider a 9th order system (C,A,B) given in Fig. 1. Elements of matrices C, A and B are pseudo-random numbers with absolute values in the interval $[1,10]$. The set Λ_n is chosen similarly, Fig. 2. Fig. 2 also contains the spectrum $\sigma(A)$ as well as coefficients f_{d_i} of the polynomial $f_d(s)$ associated with the given set Λ_n. Elements of initial guess K_{in} for the matrix

K are pseudo-random numbers satisfying $\|A\| \approx \|B K_{in} C\|$. For the stopping condition e_0 the quantity $e_0 = 0.5 \cdot 10^{-4}$ is selected. Using the single precision arithmetic (on the PDP-11/34) after 35 iterations, the solution for the matrix K is obtained. Fig. 3 contains the closed loop matrix A + B K C, its spectrum $\sigma(A+BKC)$ and the solution K. Note that although the usual condition $m+k > n$ has not been satisfied in the treated example, the "e_0-exact" solution is obtained.

CONCLUSION

The paper presents a simple and easily applicable algorithm for solving the output feedback pole placement problem. The basic advantage is that the algorithm gives satisfactory results even when other available algorithms fail. Moreover, the algorithm being based on the priciples presented in the introduction, generalizes the standard pole placement problem.

Consequently, with further extentions it may become a standard tool in numerical aspects of linear system theory.

REFERENCES

/1/ Kimura H., "Pole assignment by gain output feedback", IEEE trans.,vol.AC-20, pp. 509-516, 1975.

/2/ Kimura H., "A further results on the problem of pole assignment by gain output feedback", IEEE trans.,vol. AC-28, pp. 458-463, 1977.

/3/ Davison E.J. and Wang S.H., "On pole assignment in linear multivariable systems using output feedback", IEEE trans. AC-20, pp. 516-518, 1975.

/4/ Seraji H., "Pole assignment technique for mutlivariable systems using unity rank output feedback", Int. J. Control, vol. 21, pp. 945-954,1975.

/5/ N. Munro, S. Novin-Hirbod, "Pole assignment using full rank output feedback compensators", Int. J.Syst. Sc. vol. 10, No. 3, pp. 285-308, 1979.

/6/ Vardulakis A.I., "A sufficient conditions for n specified eingenvalues to be asigned under constant output feedback", IEEE trans., vol.AC-20, pp. 428-429, 1975.

/7/ Willems J.C. and Hesselink W.H., "General properties of the pole placement problem", Proc. IFAC Congress, Helsinki, pp. 1725-1729, 1978.

/8/ Bingulac S., "Calculation of derivatives of characteristic polynomials", IEEE trans., vol. AC-18, pp. 751-753, 1978.

/9/ Bingulac S., "On the calculation of the matrix transfer function", IEEE trans., vol. AC-20, pp. 134-135, 1975.

/10/ Garbow B.S. and et., "Matrix eigensystem routines - EISPACK guide extension", Springer - Verlag, Lecture notes in Computer science, vol. 51, 1977.

/11/ Golub G.H. and Reinsch C., "Singular value decomposition and least square solutions", Numeric. Math., vol. 14, pp. 403-429, 1970.

APPENDIX A

Calculation of the partial derivative matrix $(\partial f/\partial K)$

In /8/ it has been shown that the partial derivatives

$$g_{ij}^{h} = \partial f_{h}/\partial k_{ij}$$

of the h-th coefficient f_h of the charachteristic polynomial f(s) of the realization (C, A+BKC, B) in respect to elements k_{ij} of the output feedback matrix $K = (k_{ij})_m^k$ are given by

$$g_{ij}^{h} = -q_{ij}^{h}\ , \quad h \in [1,n] \quad , \ i \in [1,m] \quad , \ j \in [1,k] \ , \tag{A.1}$$

The quantities q_{ij}^{h} are coefficients of the polynomial matrix

$$Q(s) = (q_{ij}(s))_m^k \quad , \quad q_{ij}(s) = \sum_{h=1}^{n} \quad q_{ij}^{h} \cdot s^{h-1} \tag{A.2}$$

appearing in the numerator of the transfer function matrix

$$W(s) = Q(s)/f(s) = B^T \left[I_n s - (A+BKC)^T \right]^{-1} C^T$$

of the realization $(B^T, (A+BKC)^T, C^T)$ which is dual to the realization (C, (A+BKC), B). The coefficients q_{ij}^{h} of the polynomial $q_{ij}(s)$ may be arranged in the n-dimensional column vector q_{ij} as

$$q^{ij} = \left[q_{ij}^{1} \quad q_{ij}^{2} \quad \cdots\cdots \quad q_{ij}^{n} \right]^T \tag{A.3}$$

Thus, according to A.1 - A.3 the matrix $(\partial f/\partial K)$ required in the step 2 of the PP algorithm, may be expressed by

$$(\partial f/\partial K) = -\left[q^{11} \,\vdots\, q^{21} \,\vdots\, \cdots \,\vdots\, q^{m1} \,\vdots\, q^{12} \,\vdots\, q^{22} \,\vdots\, \cdots\cdots \,\vdots\, q^{1k} \,\vdots\, q^{2k} \,\vdots\, \cdots \,\vdots\, q^{mk} \right] (n \times mk)$$

The above result permits obtaining the matrix $(\partial f/\partial K)$ by a straightforward application of the Souriu-Frame-Faddeev (SFF) algorithm /9/. Since the vector f(K), required by the step 2 is also calculated by the SFF algorithm, it follows that the calculation of the matrix $(\partial f/\partial K)$ does not require any additional computational effort.

APPENDIX B

Calculation of the coefficients of the characteristic polynomial of cyclic matrices

The step 4 of the stabilizing procedure SP requires calculation of coefficients of characteristic polynomial of the matrix A+BKC = H. Assuming that the matrix H is "almost" always cyclic, an alternate computational procedure, more advantageous than the SFF algorithm may be suggested. Consider the (n x n+1) matrix P given by

$$P = \left[g \,\vdots\, H \right]$$

where for elements of the n-dimensional vector g arbitrary pseudo-random numbers, satisfying $\|g\| \approx \|H\|$ may be selected. Since the pair (H,g) is almost always controllable, there are similarity operations which transform the matrix P into $\tilde{P}$ where

$$\tilde{P} = [I_n \mid p] \qquad (n \times n+1)$$

Obviously, the last column of the matrix P satisfied the conditon $f = -p$, f being the vector containing the coefficients of the characteristic polynomial of the matrix H.

The decribed procedure for calculating the vector f is computationally advantageous over the SFF algorithm since the above procedure requires only $2n^3$ multiplications, while the SFF algorithm requires approximately n^4 multiplications.

APPENDIX C

On the solution of system of linear algebraic equations

Consider a system of linear algebraic equations

$$A\,x = b \qquad (C.1)$$

where $A \in R^{n\times h}$, $x \in R^h$ and $b \in R^n$; $h = mk$

In the step 4 of the PP algorithm the system (C.1) is solved by the two following alternate approaches.

The first approach is based on the least square method /10/ implemented by the singular value decomposition. The corresponding routine may be found in /11/. As it is known, in solving system C.1 two cases may occur:

Case a: rank A = rank $[A \mid b]$

In this case an exact solution exists. If there are more than one solution the routine gives the one with the minimum norm.

Case b: rank $A <$ rank $[A \mid b]$

In this case the exact solution does not exist, but the routine yields the solution x which minimizes the norm $\|Ax - b\|$.

Consequently, utilizing this approach, the PP algorithm will either give:

- the exact solution, if it exists, or
- the approximate solution K which minimizes the quantity $\|f_d - f\|$, where f is the vector containing the coefficients of the characteristic polynomial of the closed-loop matrix A+BKC.

From this discussion it is obvious the that the described PP algorithm is applicable even when $mk < n$ which apparently is not the case with most of available pole placement algorithms.

The second approach in solving the system C.1 requires $h \geqslant n$ and rank $A = n$. In this case it is necessary from A to select n linearly independent columns. This "selection" procedure may be expressed by post multilying the matrix A by a $(h \times n)$ "selector" matrix S satisying $S^T S = I_n$. In the matrix S there are n rows which are equal to the rows of $(n \times n)$

unity matrix I_n, while the remaining rows are rows with zero elements. Then, if rank (A S) = n, the solution vector is given by

$$x = S\ (A\ S)^{-1}\ b \tag{C.2}$$

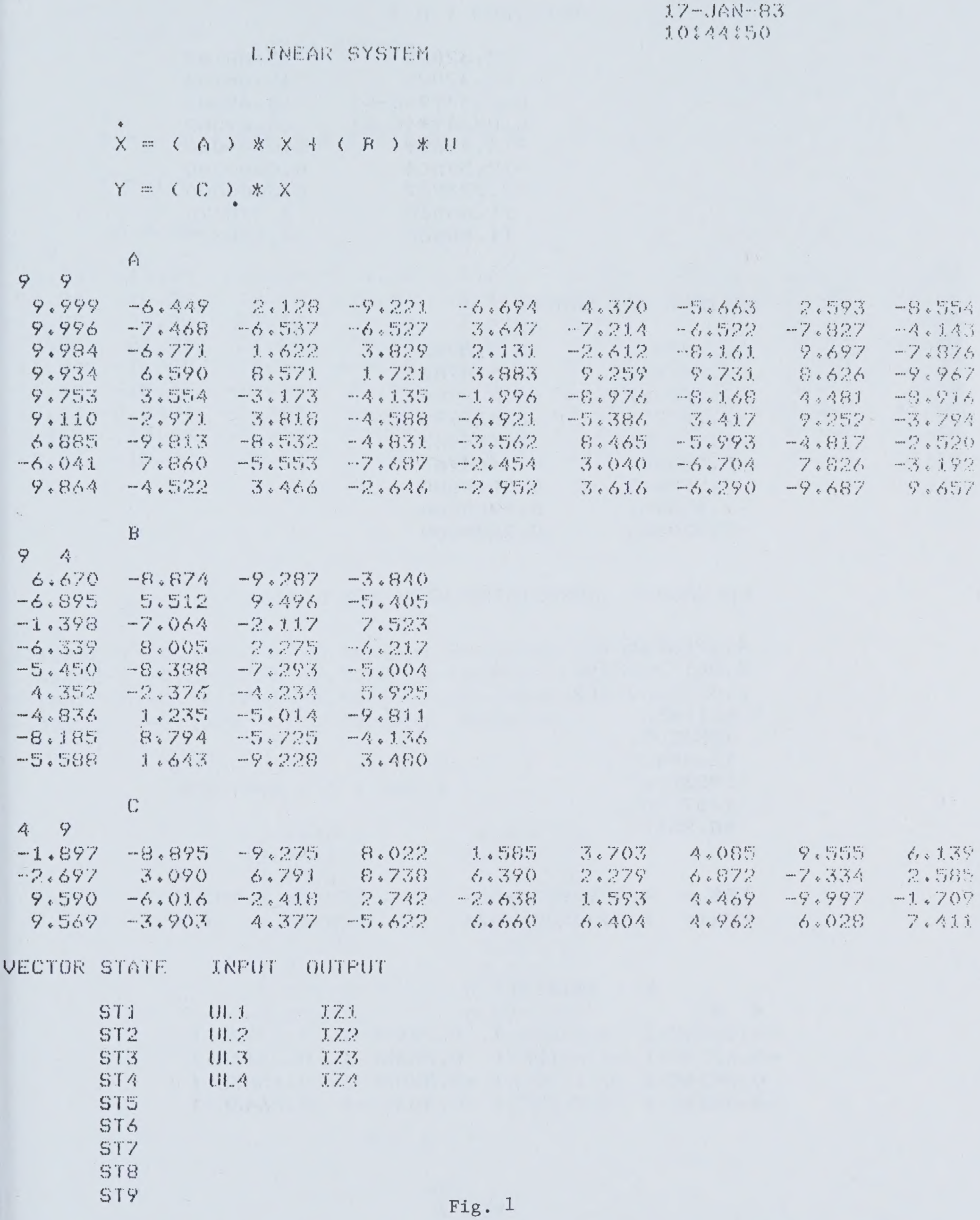
In the vector x there are only n nonzero quantities. If this approach of solving the system C.1 is incorporated into the step 4 of the PP algorithm, the selection of the set of n indices of linearly independent columns of the matrix A may be either:

- changed arbirtrary from iteration to iteration, or
- kept unchanged during the whole iterative process.

In the former case the matrix K will be of the full rank. Also, all elements of the matrix K equally contribute to the pole placement, which usually leads to the condition that absolute values of all elements k_{ij} are relatively close to each other (see Fig. 3 of the computational example). The latter case may be used in special applications as:

- obtaining a non-full rank solution matrix K, or
- achiving the pole placement by adjusting only a selected subset of the m x k solution matrix K.

Obviously, the incorporation and selection of either of the described approaches in the PP algorithm is straightforward. Moreover, if appropriate, the approach in solving the system of linear algebraic equations may be changed during the iterative process.



```
                                                          17-JAN-83
                                                          10:44:50
                 LINEAR SYSTEM


       .
       X = ( A ) * X + ( B ) * U

       Y = ( C ) * X


            A
 9   9
  9.999  -6.449   2.128  -9.221  -6.694   4.370  -5.663   2.593  -8.554
  9.996  -7.468  -6.537  -6.527   3.647  -7.214  -6.522  -7.827  -4.143
  9.984  -6.771   1.622   3.829   2.131  -2.612  -8.161   9.697  -7.876
  9.934   6.590   8.571   1.721   3.883   9.259   9.731   8.626  -9.967
  9.753   3.554  -3.173  -4.135  -1.996  -8.976  -8.168   4.481  -8.916
  9.110  -2.971   3.818  -4.588  -6.921  -5.386   3.417   9.252  -3.794
  6.885  -9.813  -8.532  -4.831  -3.562   8.465  -5.993  -4.817  -2.520
 -6.041   7.860  -5.553  -7.687  -2.454   3.040  -6.704   7.826  -3.192
  9.864  -4.522   3.466  -2.646  -2.952   3.616  -6.290  -9.687   9.657

            B
 9   4
  6.670  -8.874  -9.287  -3.840
 -6.895   5.512   9.496  -5.405
 -1.398  -7.064  -2.117   7.523
 -6.339   8.005   2.275  -6.217
 -5.450  -8.388  -7.293  -5.004
  4.352  -2.376  -4.234   5.925
 -4.836   1.235  -5.014  -9.811
 -8.185   8.794  -5.725  -4.136
 -5.588   1.643  -9.228   3.480

            C
 4   9
 -1.897  -8.895  -9.275   8.022   1.585   3.703   4.085   9.555   6.139
 -2.697   3.090   6.791   8.738   6.390   2.279   6.872  -7.334   2.585
  9.590  -6.016  -2.418   2.742  -2.638   1.593   4.469  -9.997  -1.709
  9.569  -3.903   4.377  -5.622   6.660   6.404   4.962   6.028   7.411

VECTOR STATE   INPUT   OUTPUT

        ST1       UL1      IZ1
        ST2       UL2      IZ2
        ST3       UL3      IZ3
        ST4       UL4      IZ4
        ST5
        ST6
        ST7
        ST8
        ST9
```

Fig. 1

```
                         SPECTRUM ( A )

                              12.32025            15.88183
                              12.32025           -15.88183
                            0.8834994E-01         14.69502
                            0.8834994E-01        -14.69502
                             -16.47699            0.0000000
                             -12.59564            0.0000000
                             -7.773927            0.0000000
                              11.00568             3.478375
                              11.00568            -3.478375


        DESIRED SET LAMBDA

          -7.607443        3.510783
          -7.607443       -3.510783
          -2.762251       -6.822999
          -2.762251        6.822999
          -5.923486        7.464963
          -5.923486       -7.464963
          -2.995042        0.0000000
          -7.953961        0.0000000
          -5.320856        0.0000000


          POLYNOMIAL ASSOCIATED WITH SET LAMBDA

          4.3783760E+07
          4.8017740E+07
          2.3056160E+07
           6619656.
           1284502.
           176494.8
           17232.67
           1157.205
           48.85622


          EPS =   4.9999999E-05          ITZ =      100
          EPS1=   1.0000000E-07          MOD =        1


                K - INITIAL
         4   4
       -0.2050E-1  0.1469E-1 -0.2190E-1  0.2392E-1
       -0.6772E-1 -0.4512E-1  0.2635E-1 -0.6364E-1
        0.5424E-1  0.1120E-1 -0.5880E-1 -0.4505E-1
       -0.3120E-1  0.5915E-1 -0.3832E-1  0.2643E-1
```

Fig. 2

```
                                                              17-JAN-83
                                                              10:47:38
                      LINEAR SYSTEM


         .
         X = ( A + BKC ) * X + ( B ) * U

         Y = ( C ) * X

           A + BKC
9 . 9
  12.03  -13.95  -10.40  -15.30  -14.21   2.850  -12.11   14.27  -9.639
  34.70   10.38   19.11  -45.57  -3.474  -14.33  -19.28  -31.75  -20.68
 -15.23  -22.73  -23.67   38.59   7.640   2.899   1.240   35.29   7.027
  27.83   21.21   27.00  -30.93  -3.868   2.044  -3.198  -8.240  -24.48
  11.84  -1.000  -18.94  -24.56  -17.70  -16.16  -26.57   22.58  -16.29
 -11.40  -15.47  -12.84   30.09   1.906   1.408   16.81   24.57   10.56
  26.23 -0.3484  -1.488  -46.93  -20.12  -1.864  -28.09  -9.969  -19.95
 -5.495   13.68  -4.346  -21.26  -9.278  -2.766  -17.25   8.554  -10.46
 -14.94  -15.80  -20.13   22.83  -2.047   4.952  -4.034   15.13   19.16



            K
 4   4
 0.1669E-1 -0.8303E-1 -0.4154E-1 -0.2939E-1
 0.4663E-1 -0.8756E-1 -0.7662E-2 -0.9191E-2
 0.1461    -0.1612E-1 -0.4947E-1 -0.6280E-1
-0.3182    -0.2078     0.7974E-1  0.9931E-1


                SPECTRUM ( A + BKC )

                       -2.764817           6.819299
                       -2.764817          -6.819299
                       -5.922907           7.457933
                       -5.922907          -7.457933
                       -2.934476           0.0000000
                       -7.565320           3.485707
                       -7.565320          -3.485707
                       -5.579187           0.0000000
                       -7.836512           0.0000000
```

Fig. 3

IMPACT OF EXPLICIT AND IMPLICIT CONTROL SHARING ON THE PERFORMANCE OF TWO-PERSON ONE-ACT LQG NASH GAMES

MARTIN TU and GEORGE P. PAPAVASSILOPOULOS
Department of Electrical Engineering - Systems
University of Southern California
Los Angeles, California 90089

ABSTRACT

We study the impact of explicit and implicit control sharing on the Nash performance of a two-person, one-act LQG game. The relationship between the Nash solutions under explicit and implicit control sharing is pointed out and an ordering of the uncountably many linear Nash solutions under explicit control sharing is made.

I. PROBLEM FORMULATION

x, w_1 and w_2 are independent gaussian random variables with zero mean and unit variance. There are two players denoted by 1 and 2 whose cost functionals are

$$J_1(\gamma_1, \gamma_2) = E\{q_1(x+b_1u_1+b_2u_2)^2+u_1^2+p_{12}u_2^2\} \tag{1a}$$

$$J_2(\gamma_1, \gamma_2) = E\{q_2(x+b_1u_1+b_2u_2)^2+p_{21}u_1^2+u_2^2\} \tag{1b}$$

q_1, q_2, p_{12} and p_{21} are nonnegative real numbers, and u_i is the control variable of player i. Consider the measurements

$$y_1 = h_1x + w_1 \tag{2}$$

$$y_2 = h_2x + du_1 + w_2 \tag{3}$$

$$y_2^0 = h_2x + w_2 \tag{4}$$

u_i is chosen as $r_i(\eta_i)$ where $\eta_1 = y_1$ and $\eta_2 \subset \{y_1, y_2, y_2^0, u_1\}$. We will consider the following cases:

Case A: ICS (Implicit Control Sharing) $\eta_2^A = \{y_1, y_2\}$

Case B: ECS (Explicit Control Sharing) $\eta_2^B = \{y_1, y_2^0, u_1\}$

Case C: SIS (Static Information Structure) $\eta_2^C = \{y_1, y_2^0\}$

In case of ECS, a particular Nash solution (γ_1^*, γ_2^*) is called a Stackelberg solution with player 1 as the leader if (i) γ_1^* satisfies the following inequality

$$\sup_{\gamma_2\in R_2(\gamma_1^*)} J_1(\gamma_1^*, \gamma_2) \leq \sup_{\gamma_2\in R_2(\gamma_1)} J_1(\gamma_1, \gamma_2) \tag{5}$$

for every $\gamma_1 \in \Gamma_1$ and (ii) $\gamma_2^* \in R_2(\gamma_1^*)$. where $R_2(\gamma_1)$ is defined by

$$R_2(\gamma_1) = \{\gamma_2^0 \in \Gamma_2 \,|\, J_2(\gamma_1, \gamma_2^0) \leq J_2(\gamma_1, \gamma_2), \ \forall \gamma_2 \in \Gamma_2\} \tag{6}$$

Our aim is to study the impact of control sharing by comparing the Nash costs $J_1(\gamma_1, \gamma_2)$ of case A and B (including the Stackelberg cost) with that of case C.

Nash Solution of Case A. If

This research was supported by AFOSR Grant 82-0174 and by the USC Faculty Research and Innovation Fund.

$$1+q_1b_1^2+q_2b_2^2+\zeta_A(q_1-p_{12})b_1b_2 \neq 0 \tag{7}$$

then the unique Nash solution of Case A is given by

$$\gamma_{1A}(y_1) = -\{1+q_1b_1^2+q_2b_2^2+\zeta_A(q_1-p_{12}q_2)b_1b_2\}^{-1}\{b_1q_1+\zeta_A(q_1-p_{12}q_2)b_2\}h_1(1+h_1^2)^{-1}y_1 \tag{8a}$$

$$\gamma_{2A}(y_1,y_2) = -(1+q_2b_2^2)^{-1}q_2b_2\{b_1\gamma_{1A}(y_1)+[h_1y_1+h_2(y_2-d\gamma_{1A}(y_1)](1+h_1^2+h_2^2)^{-1}\} \tag{8b}$$

where

$$\zeta_A = -(1+q_2b_2^2)^{-1}q_2b_2h_2d(1+h_1^2+h_2^2)^{-1} \tag{9}$$

Nash Solutions of Case B (Linear Class). There exist uncountably many linear Nash solutions given by

$$\gamma_{1B}(y_1) = -\{1+q_1b_1^2+q_2b_2^2+\zeta(q_1-p_{12}q_2)b_1b_2\}^{-1}\cdot\{b_1q_1+\zeta(q_1-p_{12}q_2)b_2\}h_1(1+h_1^2)^{-1}y_1 \tag{10a}$$

$$\gamma_{2B}(y_1,y_2^0,u_1) = -(1+q_2b_2^2)^{-1}q_2b_2\{b_1\gamma_{1B}(y_1)+[h_1y_1+h_2y_2^0](1+h_1^2+h_2^2)^{-1}\}+\zeta(u_1-\gamma_{1B}(y_1)) \tag{10b}$$

where ζ is any real number such that

$$1+q_1b_1^2+q_2^2b_2^2+\zeta(q_1-p_{12}q_2)b_1b_2 \neq 0 \tag{11}$$

Let us call L, the class of all these linear solutions (10).

Stackelberg Solution of Case B. The Stackelberg solution with player 1 as the leader is denoted by $(\gamma_{1s},\gamma_{2s})$ and

$$(\gamma_{1s},\gamma_{2s}) = (\gamma_{1B},\gamma_{2B})_{\zeta=-(1+q_2b_2^2)^{-1}q_2b_1b_2} \tag{12}$$

Nash Solution of Case C. The Nash solution of Case C is a special one of CaseA with $\zeta_A=0$ in (9). It is also a special one of Case B with $\zeta=0$ in (11). (For a rigorous way of finding these solutions see [4-6].)

Notice that the Nash solution of Case A depends on ζ_A which in turn depends on d. To different $d\in R$ in (3), there corresponds different $(\gamma_{1A},\gamma_{2A})_d$ of (8) provided that (7) holds. Let us call M, the class of all these solutions $(\gamma_{1A},\gamma_{2A})_d$.

II. AN ORDERING OF THE NASH SOLUTIONS

Consider the Nash pair (10), with

$$\zeta=\zeta_A= -(1+q_2b_2^2)^{-1}q_2b_2h_2d\,(1+h_1^2+h_2^2)^{-1} \tag{13}$$

((11) is satisfied), then $(\gamma_{1A},\gamma_{2A})$ and $(\gamma_{1B},\gamma_{2B})$ are equivalent in the sense that they represent the same random variable pair. Hence we write

$$(\gamma_{1A},\gamma_{2A})_d = (\gamma_{1B},\gamma_{2B})_{\zeta=\zeta_A} \tag{14}$$

If $q_2h_2\neq 0$, then to each Nash pair in L with some particular ζ, there corresponds a Nash pair in M with some particular d in (3) and vice versa. We thus have the following lemma.

Lemma 1: There is a 1-1 correspondence between M and L where

$$M \triangleq \{(\gamma_{1A},\gamma_{2A})_d \mid d\in R \text{ such that (7) holds}\} \tag{15}$$

$$L \triangleq \{(\gamma_{1B},\gamma_{2B})_\zeta \mid \zeta\in R \text{ such that (11) holds}\} \tag{16}$$

and the correspondence is given by (14).

Substituting $(\gamma_{1B},\gamma_{2B})_\zeta$ of (10) into (1a), we obtain $J_1(\gamma_{1B},\gamma_{2B})$ as a differentiable function of ζ except possibly at some singular point g where

$$1+q_1b_1^2+q_2b_2^2+g(q_1-p_{12}q_2)b_1b_2 = 0 \tag{17}$$

It holds

$$\frac{d}{d\zeta}J_1(\gamma_{1B},\gamma_{2B}) = \left\{\frac{b_2^2h_1^2(q_1-p_{12}q_2)^2(1+q_2b_2^2)}{(1+h_1^2)[1+q_1b_1^2+q_2b_2^2+\zeta(q_1-p_{12}q_2)b_1b_2]^3}\right\}\cdot\left(\zeta+\frac{q_2b_1b_2}{1+q_2b_2^2}\right) \tag{18}$$

Let b_1, b_2 and $(q_1-p_{12}q_2)$ be positive-valued, then:

$$\frac{d}{d\zeta}J_1(\gamma_{1B},\gamma_{2B}) \begin{cases} >0 & \text{if } \zeta > -(1+q_2b_2^2)^{-1}q_2b_2b_1, \text{ or } \zeta<g \\ =0 & \text{if } \zeta = -(1+q_2b_2^2)^{-1}q_2b_2b_1 \\ <0 & \text{if } g<\zeta< -(1+q_2b_2^2)^{-1}q_2b_2b_1 \end{cases} \tag{19}$$

Several important features of the impact of ECS and ICS on player 1's Nash cost are summarized in the following theorem.

Theorem 1: (i) Among L, the set of uncountably many linear Nash solutions under ECS, the unique local and thus global, minimum of J_1 is given by $(\gamma_{1B},\gamma_{2B})_{\zeta=-(1+q_2b_2^2)^{-1}q_2b_2b_1}$ which is the Stackelberg solution.

(ii) Under ECS, player 1 can do better than under SIS if $\zeta\in[-(1+q_2b_2^2)^{-1}q_2b_2b_1, 0)$.

(iii) Under ICS, player 1 can do better than under SIS if $d\in(0, b_1(1+h_1^2+h_2^2)/h_2]$.

Remark 1: Theorem 1 shows that Uchida's claim (the SIS solution is a local minimum of J_1 among L, Remark 3.3(i) of [2]) is false.

Remark 2: Theorem 1 indicates that the Stackelberg solution is beneficial to player 1 than the SIS Nash solution, which was previously pointed out by Simaan and Cruz [3]. It is not difficult to see that the Nash solution under ECS considered in Theorem 2 of [1] is actually a Stackelberg solution.

Remark 3: The parameter d in (3) can be regarded as a measure more with which player 1 shares his control implicitly to player 2. In a leader-follower situation, if the leader cannot transmit his control value to the follower free from noise, then by designing $d=(1+h_1^2+h_2^2)b_1/h_2$ in (3) and playing Nash, the leader can expect the same performance as in a Stackelberg game where the follower has perfect knowledge of the leader's control value.

REFERENCES

1. Y.C.Ho, I.Blau and T.Basar, "A Tale of Four Information Structures," Lecture Notes in Economics and Mathematical Systems, Vol. 107, Springer-Verlag, 1975.
2. K.Uchida, "On Nash Solutions in a Nonzero Sum Game with Partially Nested Information Structures," Large Scale System 3 (1982).
3. M. Simaan and J.B. Cruz, Jr., "On the Stackelberg Strategy in Nonzero Sum Games," Journal of Optimization Theory and Applications, Vol. II, No. 5, 1973.
4. G.P. Papavassilopoulos, "On the Linear-Quadratic-Gaussian Nash Game with One-Step Delay Observation Sharing Pattern," IEEE AC-27, No. 5, 1982.
5. T. Basar, "Decentralized Multicriteria Optimization of Linear Stochastic Systems," IEEE AC-23, No. 2, 1978.
6. T. Basar, "Hierarchical Decision Making Under Uncertainty," Proceedings of the 1978 Kingston Conference.

REDUCED-ORDER CONTROL FOR DISCRETE-TIME SYSTEMS WITH SLOW AND FAST MODES

MINH T. TRAN, Boeing Co., 3801 S. Oliver, Wichita, KS 67210
MAHMOUD E. SAWAN, EE Dept., Wichita State Univ., Wichita, KS 67208

SUMMARY

Linear shift invariant discrete-time systems satisfying the two-time-scale property are considered. In particular, a reduced-order control as a function of the slow states alone is designed from a given stabilizing control which is a function of both slow and fast states. This reduced-order control will provide a first-order approximation of the original quadratic performance index, which is not necessarily minimized by the given stabilizing full-order control.

SLOW AND FAST SUBSYSTEM FORMULATION

Consider the linear shift invariant discrete-time system

$$X_1(n+1) = (I + \alpha A_{11})X_1(n) + \alpha A_{12}X_2(n) + \alpha B_1 U(n), \; X_1(0) = X_{10} \qquad (1a)$$

$$X_2(n+1) = A_{21}X_1(n) + A_{22}X_2(n) + B_2 U(n), \; X_2(0) = X_{20} \qquad (1b)$$

where $X_1 \varepsilon R^{n_1}$ are the slow modes, $X_2 \varepsilon R^{n_2}$ are the fast modes, $U \varepsilon R^{m}$ is the input, α is a small positive scalar and A_{11}, A_{12}, A_{21}, A_{22}, B_1 and B_2 are constant matrices with appropriate dimensions. This system possesses the two-time-scale property, that is, it has n_1 eigenvalues with magnitude in the neighborhood of unity, and n_2 eigenvalues with magnitudes clustering around the origin of the unit circle.

Assume that we have a feedback control

$$U(n) = G_1 X_1(n) + G_2 X_2(n) \qquad (2)$$

which will be applied to system (1). It is necessary that this control will result in a stable system when being applied to system (1) even though such G_1 and G_2 are not guaranteed to exist. However, conditions under which they can be designed from slow and fast subsystem performance specifications were obtained earlier [2]. Here we assume that G_1 and G_2 are given. Applying the control (2) to system (1), we have

$$X_1(n+1) = [I + \alpha(A_{11} + B_1 G_1)]X_1(n) + \alpha[A_{12} + B_1 G_2]X_2(n) \qquad (3a)$$

$$X_2(n+1) = [A_{21} + B_2 G_1]X_1(n) + [A_{22} + B_2 G_2]X_2(n) \qquad (3b)$$

As shown in [1,2], we can obtain the slow subsystem as

$$\frac{dX_s(t)}{dt} = A_0 X_s(t) \quad , \quad X_s(0) = X_{10} \qquad (4)$$

where

$$A_0 = A_{11} + B_1 G_1 + (A_{12} + B_1 G_2)(I - A_{22} - B_2 G_2)^{-1}(A_{21} + B_2 G_1)$$

and the fast subsystem as

$$X_f(n+1) = (A_{22} + B_2 G_2)X_f(n) \quad , \quad X_f(0) = X_{20} - \overline{X}_2(0) \qquad (5)$$

APPROXIMATE CONTROL DEVELOPMENT

Consider the performance index

$$J = \tfrac{1}{2} \alpha \sum_{n=0}^{\infty} X^T(n)QX(n) + U^T(n)RU(n) \tag{6}$$

where

$$Q = \begin{bmatrix} Q_1 & Q_2 \\ Q_2^T & Q_3 \end{bmatrix} \geqq 0 \quad , \quad R > 0 \quad , \quad X(n) = \begin{bmatrix} X_1(n) \\ X_2(n) \end{bmatrix}$$

It is desired to find a control as a function of the slow state X_1 alone which will yield a close approximation of the performance of that of the control (2) when being applied to system (1). To obtain this reduced-order control, we let $\alpha = 0$ and get the expression for X_2 which is then substituted into (2) to get

$$U_r = [G_1 + G_2(I - A_{22} - B_2G_2)^{-1}(A_{21} + B_2G_1)]X_1 = F_1X_1 \tag{7}$$

Letting

$$A = \begin{bmatrix} I + \alpha A_{11} & \alpha A_{12} \\ A_{21} & A_{22} \end{bmatrix} \quad , \quad B = \begin{bmatrix} \alpha B_1 \\ B_2 \end{bmatrix}$$

$$G = [\; G_1 \quad G_2 \;] \quad , \quad F = [\; F_1 \quad 0 \;]$$

and assuming that $(A_{22} + B_2G_2)$, A_0, $(A + BG)$ and $(A + BF)$ are stable, we have the following result.

Main Result: Let J and J_r be the performance costs of the system when (2) and (7) are being applied to (1), respectively, then we have

$$J_r = J + 0(\alpha) \tag{8}$$

This result can be obtained by showing that the performance costs are given in terms of the solutions of the Lyapunov equations which can be expressed in a power series expansion at $\alpha = 0$. Partitioning these Lyapunov equations and evaluating at $\alpha = 0$, their difference can be shown to be equal to zero which yields the result.

REFERENCES

[1] Blankenship G., "Singularly Perturbed Difference Equations in Optimal Control Problem," IEEE Trans. Automat. Contr., Vol. AC-26, No. 4, Aug. 1981.

[2] Litkouhi B. and Khalil H. K., "Infinite-Time Regulators for Singularly Perturbed Difference Equations," Proc. of 20th Allerton Conf. on Communication, Computers and Control, University of Illinois, Urbana, 1982.

[3] Tran M. T., Sawan M. E., "Reduced-Order Discrete-Time Models," International Journal of Systems Science, Vol. 14, No. 7, 1983.

ON A REACTIVE POWER FAULT MODELING IN LARGE SCALE ELECTRIC POWER SYSTEMS

Marija Ilic-Spong* and Matthew Varghese
Cornell University
School of Electrical Engineering
Ithaca, NY 14853

ABSTRACT

The reactive power problem of an electric power system is concerned with the mutual dependence of the reactive power flows $\underline{Q}$ and bus voltage magnitudes $\underline{V}$, assuming that the bus voltage phase angles $\underline{\theta}$ are known parameters. In the first part of this paper we suggest some answers to the generator performance under disturbances in the reactive power-voltage Q-V network. A linearized formula is derived for computing changes in the reactive power injection of generators, which are subject only to the primary voltage control, due to load outages. In the second part of this work modeling of transmission line outages is studied on the Q-V network. We derive linearized formulae for reactive power distribution factors to measure changes in the reactive power flow in line r-s due to a change of reactive power flow in line p-q, when the line p-q is subject to an outage, for any buses p, q, r and s in the given Q-V network. These results are constant, i.e. disturbance and/or operating point independent and can be used for fast estimates of changes in the reactive power flows due to different line outages, without performing time comsuming load flow calculations.

I. INTRODUCTION

The control of voltages and reactive power generation and flows long recognized as an important problem in power system operation has taken on new importance following blackouts in Western Europe caused by the phenomenon of voltage collapse [1]. In spite of the importance of reactive power control many theoretical issues are still unsolved, in particular the formulation of the problem as a large scale and system theoretic one. On the other hand, many results exist for the active power generation and flows modeling and control and it is clear, based on that work, what the possible open problems on the reactive power issue are. Note that it is possible to analyze active power and reactive power problems separately, as long as the commonly used decoupling assumption [1] is accepted. Therefore we mean here under the active power problem the analysis of the relations between the active power flows $\underline{P}$ and bus voltage phase angles $\underline{\theta}$, under the assumption that the bus voltage magnitudes $\underline{V}$ are known parameters. The reactive power problem is concerned with the mutual dependence of the reactive power flows $\underline{Q}$ and bus voltage magnitudes $\underline{V}$, assuming that the bus voltage phase angles $\underline{\theta}$ are known parameters. An estimate of the error due to the decoupling assumption has recently been studied in [2]. Throughout this paper we make this assumption.

A qualitative difference in simulating outages for the active and reactive power networks lies in the fact that in the P-θ network generators tend not to react to the fault unless otherwise controlled (PV buses), i.e. their injected active power will not change after a network disturbance. On the other hand, in the Q-V decoupled network, generators have a natural tendency to change the reactive power generation, unless subject to some secondary level control. So called primary voltage control will tend to keep generators as PV buses, while their reactive power generation will change due to a stator current change. In simulating disturbances in the P-θ network, two approaches are taken [3]:

---

*This work was partially supported by the NSF grant ECS-8214143.

(i) An assumption is made that the slack bus [1] would produce enough active power to compensate for a disturbance. In this case the active power generation will not change at any other generator in the network.

(ii) In the case that the capacity of a slack bus is lower than the demand for change the so called participation factors technique [3] is used to compensate for the power disturbance.

In the first part of this paper we suggest some answers to the generator performance under disturbances in the reactive power network. We derive a linearized formula to compute changes in the reactive power injection of generators, which are subject only to the primary voltage control, due to load outages. In the second part of this work we have studied modeling of transmission line outages in more details [4] for the Q-V network. In particular a technique known as the current distribution factors for active power generation and flow studies [4] is adopted for the reactive power studies on the S-E graph decoupled Q-V network [5]. As explained in [4] one can see that the current distribution factors technique is not accurate enough for the reactive power flow studies. Here we develop, using similar principles as in [4], on a different graph, formulae for fast computation of changes in reactive power (transmitted and loss) due to a transmission line outage. Note that to give a network formulation to the reactive power flow redistribution it has been necessary to use the S-E graph since the power shift theorem [6] only holds on this graph. And finally, we conjecture that the vector of changes of reactive power flows (transmitted and loss) could be used as a much better criteria for estimating how far the new equilibrium is from the optimal (normal) operating point following a disturbance. We claim that due to different masking effects commonly used scalar criteria, for contingency ranking in particular [3], are much worse than the suggested vector of transmitted and loss reactive power changes. In the latter one can see, line by line, where the worst effect of the outage occurs. The reactive power distribution factors, similarly to the current distribution factors, have to be computed only once and stored for the future on-line estimates of any disturbance effects on the network.

II. GENERATOR PARTICIPATION FACTORS OF REACTIVE POWER

Automatic voltage regulators (AVR) on the generating units modify the excitation current of the machine in order to maintain voltages across the stator terminals equal to values that have been set in advance. The excitation current changes lead indirectly to changes in the reactive power output. If the reactive power outages are significantly large they cause a change of the systems load and generator voltages. As a consequence, to maintain the generator voltages constant under these conditions, a certain amount of reactive power has to be either supplied or absorbed by the generators. This control of voltage magnitudes is local and it is referred to as the primary voltage regulation. Here we show a linearized method for computing generator participation factors of reactive power due to the primary voltage regulation in reacting to a reactive power load outage. An assumption that the slack bus would pick up the major change in injection due to a disturbance is wrong, as opposed to the same assumption in the P-θ network. Instead, the following theorem states the approximate amounts of generator reactive power changes.

Theorem 1: Given a single load outage at bus j with a reactive power change ΔQ_j the participation factor of the i-th generator due to this outage is approximately equal to

$$\Delta Q_i = g_{ij} \; \Delta Q_j \tag{1}$$

Here G_{ij} is the (ij)th element of the matrix G defined as: If

$$B = \left(\frac{\partial Q}{\partial \underline{V}}\right) - \left(\frac{\partial Q}{\partial \underline{\theta}}\right) \left(\frac{\partial P}{\partial \underline{\theta}}\right)^{-1} \left(\frac{\partial P}{\partial \underline{V}}\right) \tag{2}$$

$$= \begin{bmatrix} B_{GG} & B_{GL} \\ B_{LG} & B_{LL} \end{bmatrix} \tag{3}$$

then

$$G = (B_{GL}) \, (B_{LL})^{-1} \tag{4}$$

Proof: Define $\underline{V}$ to be a vector of bus voltage magnitudes

$$\underline{V} = [\underline{V}_G \quad \underline{V}_L]^T \tag{5}$$

with $\underline{V}_G$ - vector of generator voltage magnitudes and $\underline{V}_L$ - vector of load voltage magnitudes and vector $\underline{\theta}$ to be vector of bus voltage phase angles

$$\underline{\theta} = [\underline{\theta}_G | \underline{\theta}_L]^T \tag{6}$$

with $\underline{\theta}_G$ - vector of generator voltage phase angles and $\underline{\theta}_L$ - vector of load voltage phase angles

The linearized load flow equations around the normal operating point for a power network with n buses take on the form

$$\begin{bmatrix} \Delta \underline{Q}_G \\ \Delta \underline{Q}_L \\ \Delta \underline{P}_G \\ \Delta \underline{P}_L \end{bmatrix} = \begin{bmatrix} \frac{\partial Q_G}{\partial V_G} & \frac{\partial Q_G}{\partial V_L} & \frac{\partial Q_G}{\partial \theta_G} & \frac{\partial Q_G}{\partial \theta_L} \\ \frac{\partial Q_L}{\partial V_G} & \frac{\partial Q_L}{\partial V_L} & \frac{\partial Q_L}{\partial \theta_G} & \frac{\partial Q_L}{\partial \theta_L} \\ \frac{\partial P_G}{\partial V_G} & \frac{\partial P_G}{\partial V_L} & \frac{\partial P_G}{\partial \theta_G} & \frac{\partial P_G}{\partial \theta_L} \\ \frac{\partial P_L}{\partial V_G} & \frac{\partial P_L}{\partial V_L} & \frac{\partial P_L}{\partial \theta_G} & \frac{\partial P_L}{\partial \theta_L} \end{bmatrix} \begin{bmatrix} \Delta \underline{V}_G \\ \Delta \underline{V}_L \\ \Delta \underline{\theta}_G \\ \Delta \underline{\theta}_L \end{bmatrix} \tag{7}$$

Now, if the primary voltage regulation is in effect then the following holds:

Assumption 1:

$$\Delta \underline{V}_G = \underline{0} \tag{8}$$

$$\Delta \underline{P}_G = 0 \tag{9}$$

$$\Delta \underline{P}_L = 0 \tag{10}$$

Performing Gaussian elemination on (7) under this assumption leads to

$$\begin{bmatrix} \Delta \underline{Q}_G \\ \Delta \underline{Q}_L \end{bmatrix} = \left(\left(\frac{\partial \underline{Q}}{\partial \underline{V}}\right) - \left(\frac{\partial \underline{Q}}{\partial \theta}\right) \left(\frac{\partial \underline{P}}{\partial \theta}\right)^{-1} \left(\frac{\partial \underline{P}}{\partial \underline{V}}\right) \right) \begin{bmatrix} \Delta V_G \\ \Delta V_L \end{bmatrix} \tag{11}$$

Define

$$B = \left(\frac{\partial \underline{Q}}{\partial \underline{V}}\right) - \left(\frac{\partial \underline{Q}}{\partial \theta}\right) \left(\frac{\partial \underline{P}}{\partial \theta}\right)^{-1} \left(\frac{\partial \underline{P}}{\partial \underline{V}}\right) \tag{12}$$

$$= \begin{bmatrix} B_{GG} & B_{GL} \\ B_{LG} & B_{LL} \end{bmatrix} \tag{13}$$

and since $\Delta V_G = 0$, (11) implies

$$\Delta \underline{Q}_G = B_{GL}\ \Delta \underline{V}_L \tag{14}$$

and

$$\Delta \underline{Q}_L = B_{LL}\ \Delta \underline{V}_L \tag{15}$$

By computing the vector of load bus voltage changes from (15) and replacing it in (14), we obtain the reactive power injection changes at the generator buses

$$\Delta \underline{Q}_G = B_{GL}\ B_{LL}^{-1}\ \Delta \underline{Q}_L = G\ \Delta \underline{Q}_L \tag{16}$$

For a single contingency at load j

$$\Delta \underline{Q}_L = (0 \quad 0 \quad \cdots \quad 0 \ \Delta Q_j \ 0 \quad \cdots \quad 0)^T \tag{17}$$

and from (16) the jth column of the G matrix gives the participation factors of all generators due to the jth load outage.

Corollary 1: For a multiple disturbance case one needs to store as many columns of G matrix as there are disturbances and then the reactive power change in generator i is

$$\Delta Q_{Gi} = \sum_{k \in \{\text{set of load outages}\}} g_{ik}\ \Delta Q_k \tag{18}$$

Proof: Immediate from (16) since

$$\Delta \underline{Q}_L = (\Delta Q_1 \ \Delta Q_2 \quad \cdots \quad 0 \quad \cdots \quad 0)^T \tag{19}$$

The above derivation corresponds to the primary voltage control when generators have adequate reactive power available to maintain the voltage magnitude constant. We have also performed the worst case analysis when all generators (excluding slack) hit their reactive power limits. However due to space limitations we will not present this here.

III. REACTIVE POWER DISTRIBUTION FACTORS FOR LINE OUTAGES

In this section we derive approximate (linearized) expressions for reactive power distribution factors $K_{rs,pq}$ to measure changes in the reactive power flow in line r-s due to a unit change of reactive power flow in line p-q, when the line p-q is subject to an outage, for any four nodes p, q, r and s in a given power network. These coefficients are constant, i.e. disturbance and/or operating point independent and can be used for fast estimates of changes in the reactive power flows due to different line outages, without performing time consuming load flow calculations. The motivation for this result stems from the existence of the similar formulae for the current distribution factors for line outages used on-line in monitoring the active power-phase angle P-θ network [4]. Although aimed for a similar application in the Q-V network, it would be very hard to obtain the reactive power distribution factors (RPDF) using analogous principles as in deriving the current distribution factors (CDF) [4] on the conventional current-voltage I-E graph. Namely, due to reactive power losses, the reactive power flows do not obey the flow preservation law on the I-E graph and therefore are not subject to the same algebraic constraints as the current variables (subject to Kirchhoff's current law). To circumvent this difficulty and express our results directly in terms of reactive power flow changes we use the complex power-complex voltage S-E graph [5] and its decoupled version Q-V graph [5]. The Q-V graph is derived from the S-E graph under the decoupling assumption [1]. Also we deviate from the original version of the S-E graph in the sense that because of symmetry reasons the L-section representing a transmission line in this graph is transformed into the ∩-section. Next, the linearized Q-V network is used so that the superposition principle and the Q-shift theorem, analogous to the I-shift theorem [6], are exploited for modeling reactive power line outages.

Definition 1: The ∩ subgraph of the S-E graph representing a transmission line between nodes i and j consists of three branches, one corresponding to a completely transmitted power S^T_{ij} between nodes i and j and two branches representing reactive power losses S^L_{io} and S^L_{jo} (Figure 1). The branch relations are defined as

$$S^T_{ij} = (E_i + E_j)(E_i - E_j)^* \frac{Y^*_{ij}}{2} \tag{20}$$

and

$$S^L_{io} = S^L_{jo} = |E_i - E_j|^2 \frac{Y^*_{ij}}{2} \tag{21}$$

with Y_{ij} admittance of the line i-j.

Definition 2: The ∩ subgraph of the Q-V graph representing a transmission line between nodes i and j is obtained from the ∩ subgraph on the S-E graph under the assumption that the bus voltage phase angles θ_i and θ_j are given (known) parameters (Figure 2). The Q-V branch relations are (with Re $Y_{ij} = 0$):

(i) Transmitted reactive power between nodes i and j

$$Q^T_{ij} = \mathrm{Im}\, S^T_{ij} = (V_i^2 - V_j^2) \frac{B_{ij}}{2} \tag{22}$$

and

(ii) Loss reactive powers on line i-j

$$Q^L_{i0} = Q^L_{j0} = (V_i^2 + V_j^2 - 2V_iV_j \cos\theta_{ij}) \frac{B_{ij}}{2} \tag{23}$$

where $B_{ij} = \mathrm{Im}Y^*_{ij}$ (* means complex conjugate).

Definition 3: The Q-V graph of an electric power network consists of Q-V ∩ subgraphs representing transmission elements, independent voltage sources representing generators and independent reactive power sources representing loads.

Definition 4: The linearized Q-V network has the reactive power-voltage branch relations

$$\Delta Q^T_{ij} = (V_i \Delta V_i - V_j \Delta V_j)\, B_{ij} \tag{24}$$

and

$$\Delta Q^L_{i0} = \Delta Q^L_{j0} = ((V_i - V_j \cos\theta_{ij})\, \Delta V_i + (V_j - V_i \cos\theta_{ij})\, \Delta V_j)\, B_{ij} \tag{25}$$

Next, after introducing the above necessary definitions we follow a similar route in obtaining the RPDF on the linearized Q-V network as in [4] for the CDF on the I-E graph. For this, consider a power network shown in Figure 3 and suppose that we are interested in outage effects of the line connecting nodes p and q on changes in bus voltages and reactive power injections at buses r and s connected by line r-s. The independent voltage sources $\underline{V}_G$ and independent reactive power sources $\underline{Q}_L$ stand for all generators and loads, respectively, other than loads and generators connected to the buses p and q. In this derivation on the linearized Q-V network the Q-shift theorem is used as a direct analogy to the I-shift theorem on the I-E network [6] since in these two graphs the roles of currents and power flows are interchanged. Also principle of superposition is valid here for power sources the same as for the current sources on the linear I-E network. Now, based on the Q-shift theorem power network on Figure 3 could be thought of as a network on Figure 4. The post-outage network state is obtained as a superposition of two networks, one on Figure 3 and the other network the same except that the sources 1, 2, 3 and 4 have opposite directions. Therefore, in order to obtain the post-outage state we compute the changes in network due to the sources 1, 2, 3 and 4.

On the other hand, let us write the decoupled linearized load flow equations for all loads except for possibly buses p and q if they are load buses. Since we assume that generators have sufficient capacity $\Delta\underline{V}_G = 0$ and

$$\Delta \underline{Q}_L = \left(\frac{\partial \underline{Q}_L}{\partial \underline{V}_L}\right) \Delta V_L \tag{26}$$

from which voltage changes due to reactive power changes are

$$\Delta \underline{V}_L = [X'_{LL}]\, \Delta \underline{Q}_L \tag{28}$$

Next, an outage of line p-q can be thought of as a superposition of outages of four reactive power sources Q^L_{p0}, Q^L_{q0}, Q^T_{pq} and Q^T_{pq}. Therefore, in the first step we simulate outage of both Q^T_{pq} and in the second step outage of Q^L_{po} and Q^L_{qo} ($Q^L_{po} = Q^L_{qo}$).

Step 1: $\Delta \underline{Q}_L$ is the vector of reactive power injections which has nonzero entries Q^T_{pq} and $-Q^T_{pq}$ in locations p and q. A typical set of equations resulting from the equation (26) is

$$\Delta V_r = X_{rp}' Q^T_{pq} - X_{rq}' Q^T_{pq} \tag{29}$$

$$\Delta V_s = X_{sp}' Q^T_{pq} - X_{sq}' Q^T_{pq} \tag{30}$$

where $X^T_{ij} = (X_{LL}')_{ij}$

The new voltages at buses r and s are

$$V_r' = V_r + \Delta V_r \tag{31}$$

and

$$V_s' = V_s + \Delta V_s \tag{32}$$

The effect on reactive power flow changes in line r-s is given therefore from (24) and (25) as:

(i) The effect on Q^T_{rs} is

$$\Delta Q^T_{rs} = (V_r \Delta V_r - V_s \Delta V_s)B_{rs} \tag{33}$$

and

$$\Delta Q^L_{so} = \Delta Q^L_{ro} = ((V_r - V_s \cos\theta_{rs})\,\Delta V_r + (V_s - V_r \cos\theta_{rs})\,\Delta V_s)B_{rs} \tag{34}$$

and from equations (29) and (30) combined with (33), (34) the reactive power changes are

$$\Delta Q^T_{rs} = (V_r(X_{rp}' - X_{rq}') - V_s(X_{sp}' - X_{sq}'))B_{rs}\, Q^T_{pq} \tag{35}$$

or

$$\Delta Q^T_{rs} = K^T_{rs},{}^T_{pq} \cdot Q^T_{pq} \tag{36}$$

where $K^T_{rs},{}^T_{pq}$ is the RPDF which defines how the transmitted power in line r-s is affected by the change in transmitted power in line p-q, and takes on the form

$$K^T_{rs},{}^T_{pq} = (V_r(X_{rp}' - X_{rq}') - V_s(X_{sp}' - X_{sq}'))B_{rs} \tag{37}$$

and similarly

$$\Delta Q^L_{ro} = \Delta Q^L_{so} = K^L_{rs},{}^T_{pq}\, Q^T_{pq} \tag{38}$$

with $K^L_{rs},{}^T_{pq}$ the RPDF which defines the change in the loss power through the line r-s due to the change in the transmitted reactive power in line p-q as

$$K^L_{rs},{}^T_{pq} = ((V_r - Vs\cos\theta_{rs})\,(X_{rp}' - X_{rq}') + (V_s - V_r\cos\theta_{rs})\,\cdot$$

$$\cdot\,(X_{sp}' - X_{sq}'))B_{rs} \tag{39}$$

Step 2: Since $Q^L_{po} = Q^L_{qo}$, $\Delta \underline{Q}_L$ is the bus reactive power vector with nonzero entries Q^L_{po} in locations p and q. The voltage magnitude changes at buses r and s due to $\Delta \underline{Q}_L$ are from equation (28)

$$\Delta V_r = (X_{rp}' + V_{rq}')Q^L_{po} \tag{40}$$

and

$$\Delta V_s = (X_{sp}' + X_{sq}')Q^L_{po} \tag{41}$$

with the new resulting voltages

$$V_r'' = V_r' + \Delta V_r \tag{42}$$

$$V_s'' = V_s' + \Delta V_s \tag{43}$$

and an analogous procedure as in Step 1 obtains

$$\Delta Q^T_{rs} = K^{T}_{rs},{}^{L}_{rs}\; Q^L_{po} \tag{44}$$

and

$$\Delta Q^L_{ro} = \Delta Q^L_{so} = K^{L}_{rs},{}^{L}_{pq}\; Q^L_{po} \tag{45}$$

With $K^{T}_{rs},{}^{L}_{pq} = (V_r(X_{rp}'+X_{rq}') - V_s(X_{sp}'+X_{sq}'))\, B_{rs}$ and

$$K^{L}_{rs},{}^{L}_{pq} = ((V_r - V_s\cos\theta_{rs})\,(X_{rp}'+X_{rq}') + (V_s - V_r\cos\theta_{rs})\,\cdot$$

$$\cdot\,(X_{sp}'+X_{sq}'))\, B_{rs} \tag{46}$$

The reactive power distribution factors $K^{T}_{rs},{}^{L}_{pq}$ and $K^{L}_{rs},{}^{L}_{pq}$ define the effect of changes in loss reactive power flow Q^L_{po} in line p-q on the changes in the transmitted and loss reactive power flows Q^T_{rs}, Q^L_{ro} in line r-s respectively.

The new flows in line r-s after the line outage p-q are by superposition

$$Q'^T_{rs} = Q^T_{rs} + K^{T}_{rs},{}^{T}_{pq}\, Q^T_{pq} + K^{T}_{rs},{}^{L}_{pq}\, Q^L_{po} \tag{47}$$

and

$$Q'^L_{ro} = Q'^L_{so} = Q^L_{so} + K^{L}_{rs},{}^{T}_{pq}\, Q^T_{pq} + K^{L}_{rs},{}^{L}_{pq}\, Q^L_{po} \tag{48}$$

and the new voltages are given by two successive corrections by equations (42) and (43). Finally, the changes in reactive power injections and voltages at the ends of the line subject to the outage need to be computed in a modified manner, since they are the only two buses which "see" the change in the network structures due to the line outage p-q. For these two buses since the injected reactive power into the node is

$$Q_i = V_i^2 B_{ii} - V_i \sum_{j\neq i} V_j B_{ij} \cos\theta_{ij} \tag{49}$$

$$\frac{\partial Q_i}{\partial B_{ik}} = V_i^2 - V_i V_k \cos\theta_{ik} \quad i = p,q;\; k = p,q;\; i \neq k \tag{50}$$

Different combinations are possible:

(1) If both buses p and q are PV buses, ΔV_p and ΔV_q are zero and changes in reactive power injections into these two nodes are

$$\Delta Q^{p} = \frac{\partial Q_p}{\partial B_{pq}} B_{pq} \tag{51}$$

and

$$\Delta Q_q = \frac{\partial Q_q}{\partial B_{pq}} B_{pq} \tag{52}$$

with the derivatives evaluated using (49).

(2) If bus p is a PV bus and bus q is a PQ bus there will be no change in ΔV_p but

$$\Delta Q_p = \frac{\partial Q_p}{\partial V_q} \Delta V_q + \frac{\partial Q_p}{\partial B_{pq}} B_{pq} \tag{53}$$

and also

$$\Delta Q_q = 0 = \frac{\partial Q_q}{\partial V_q} \Delta V_q + \frac{\partial Q_q}{\partial B_{pq}} B_{pq} \tag{54}$$

Solving for ΔV_q from (54) and for ΔQ_p from (53) we obtain all changes caused by line outage p-q.

(3) If buses p and q are both PQ buses then

$$\Delta Q_p = 0 = \frac{\partial Q_p}{\partial V_q} \Delta V_q + \frac{\partial Q_p}{\partial V_p} \Delta V_p + \frac{\partial Q_p}{\partial B_{pq}} B_{pq} \tag{55}$$

and

$$\Delta Q_q = 0 + \frac{\partial Q_q}{\partial V_p} \Delta V_p + \frac{\partial Q_q}{\partial V_q} \Delta V_q + \frac{\partial Q_q}{\partial B_{pq}} B_{pq} \tag{56}$$

Equations (55) and (56) form a set of two linear equations in two unknowns ΔV_p and ΔV_q whose solution gives all changes caused by the line outage p-q.

IV. CONCLUSIONS

An approximate technique for measuring the effects of the reactive power outages throughout the decoupled reactive power-voltage Q-V network is proposed. The memory requirement and computing time are very similar to the once in computing current distribution factors in the decoupled active power-phase angle decoupled network. Numerical examples on the standard IEEE examples show good accuracy of the technique and we conjecture that it would have promising applications for on-line monitoring of reactive power flows in control centers. Theoretically, a contribution is made in extending network theory results (I-shift theorem, superposition principle) to the power voltage S-E graph in terms of power flows as main variables.

V. ACKNOWLEDGEMENT

The authors would like to acknowledge several helpful discussions with Professors A. Phadke and J. Thorp in developing these results.

VI. REFERENCES

[1] B. Stott, "Review of load flow calculation methods", Proc. IEEE, vol. 62, pp. 916-292, 1974.

[2] John Kaye, F. Wu, "Analysis of Linearized Decoupled Power Flow Approximations for Steady State Security Accessment," Electronics

Research Lab., Memorandum No. UCB/ERL M82/71, Berkeley, 1982.

[3] A. Debs, A. Benson, "Security Assessment of Power Systems", ERDA 76-66, CONF-750867, Henniker, 1975.

[4] A. Phadke, "On the Contingency Evaluation of a Power System in Real Time", AEP, 1972.

[5] M. Ilic-Spong, R. Fischl, "Steady State Electric Network Theorems for Power Systems Based on the Complex Power-Complex Volgage S-E Graph", Submitted for publicaiton to the IEEE Trans. on Circuits and Systems.

[6] L. Chua, Nonlinear Networks, McGraw Hill, 1980.

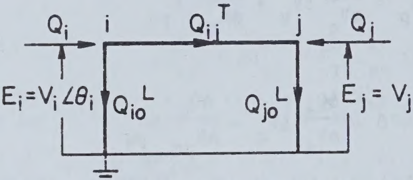

Q_i
i
Q_{ij}^T
j
Q_j
$E_i = V_i \angle\theta_i$
Q_{io}^L
Q_{jo}^L
$E_j = V_j \angle\theta_j$

FIGURE 1

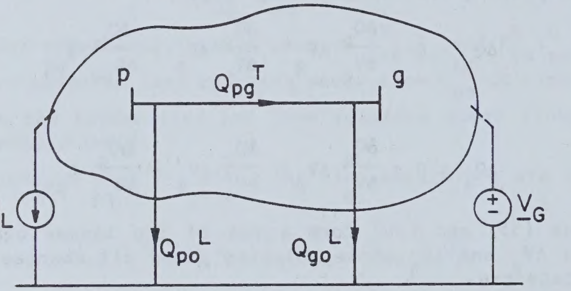

p
Q_{pg}^T
g
$\underline{Q}_L$
$\underline{V}_G$
Q_{po}^L
Q_{go}^L

FIGURE 2

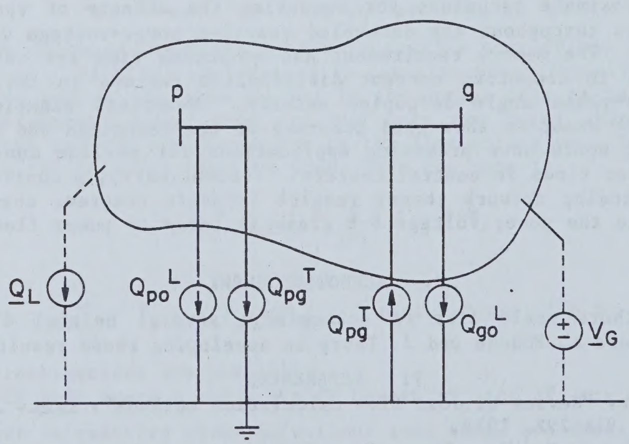

p
g
$\underline{Q}_L$
Q_{po}^L
Q_{pg}^T
Q_{pg}^T
Q_{go}^L
$\underline{V}_G$

FIGURE 3

EFFECT OF AMPLITUDE AND BANDLIMITING ON MULTI-H CPM DATA TRANSMISSION

MARK A. WICKERT AND RODGER E. ZIEMER
Electrical Engineering Department
University of Missouri-Rolla
Rolla, Missouri

ABSTRACT

The effects of transmitter filtering and transmitter amplitude limiting on the power efficiency performance of multi-h continuous-phase modulation (CPM) schemes are considered. In particular a (3/8, 4/8) linear phase code filtered with a cascade of three first order Butterworth filters at lowpass cutoff frequency of 0.55 is studied. The signal-to-noise ratio (SNR) degradation with respect to the unfiltered code at a bit error rate (BER) of 10^{-5} is 0.65dB when the receiver decision depth is nine, as opposed to 0.29dB when the decision depth is increased to sixteen. When amplitude limiting follows this filtering configuration, the BER degradation when using a decision depth of sixteen increases to 0.47dB. BER degradation results obtained using simulation techniques are also compared with minimum Euclidean distance bounds.

I. INTRODUCTION

The study of modulation schemes that are both bandwidth and power efficient is of considerable interest for use in satellite communications systems. Multi-h continuous phase modulation (CPM) is one such scheme [1], [2]. The performance of multi-h modulation in the past has been obtained primarily for an ideal additive white Gaussian noise (AWGN) channel. The optimum receiver requires a maximum likelihood sequence estimator, which is typically implemented using the Viterbi algorithm (VA). Two approaches for obtaining system bit error rate (BER) performance are through Monte Carlo simulation and bounding by using the minimum Euclidean distance.

In this paper multi-h modulation is considered in an environment that includes transmitter filtering and amplitude limiting. For a satellite communications system this would correspond to BER performance being downlink noise limited. The method of investigation used here is primarily computer simulation.

In the first section of this paper we will give an overview of full response multi-h modulation followed by a brief description of the simulation technique. System BER performance is then presented for filtering alone and filtering followed by limiting.

II. OVERVIEW OF MULTI-H

In this section we define the modulation format and then describe some of the important properties it has, including the minimum Euclidean distance.

A. Definition

We assume a constant envelope signal of the form

$$s(t,\underline{\alpha}) = \sqrt{\frac{2E}{T}} \cos(2\pi f_o t + \phi(t,\underline{\alpha})) \tag{1}$$

where E is the symbol energy, T is the symbol period, and f_o is the carrier frequency. The term $\phi(t,\underline{\alpha})$ is a modulation induced phase function of the form

$$\phi(t,\underline{\alpha}) = 2\pi \sum_{i=-\infty}^{\infty} h_i \alpha_i q(t-iT) \tag{2}$$

where here $\underline{\alpha} = \ldots \alpha_{-1}\alpha_0\alpha_1 \ldots$ is a random binary data sequence, h_i is a modulation index, and q(t) is the phase response which is the integral of a frequency pulse g(t). In general, g(t) is LT seconds in length and each α_i is chosen from an M-ary alphabet. The case considered here is known as binary full response multi-h [2] in which case L = 1. Each modulation index h_i is cyclically chosen from a set H_k of k rational numbers with common denominator

$$H_k = \left\{ h_1, h_2, \ldots h_k \right\} = \left\{ \frac{m_1}{q}, \frac{m_2}{q}, \ldots, \frac{m_k}{q} \right\} \tag{3}$$

with

$$h_{i+k} = h_i \text{ for } i = -1, 0, 1, \ldots$$

and each m_i is greater than zero and less than q.

The type of frequency pulse shape considered in this paper is a linear phase or rectangular (REC) frequency pulse:

$$g(t) = \begin{cases} \frac{1}{2T}, & 0 \leq t \leq T \\ 0, & \text{elsewhere} \end{cases} \tag{4}$$

For the linear phase pulse with $H_1 = \{\frac{1}{2}\}$ the modulation scheme reduces to ordinary MSK. The frequency pulse is normalized so that

$$\int_o^T g(t)dt = \tfrac{1}{2} = q(T) \tag{5}$$

An instructive way of displaying the possible phase trajectories of a particular multi-h code is with a trellis diagram, which is shown in Figure 1 for the linear phase code $H_2 = \{3/8, 4/8\}$. The trellis consists of all possible phase trajectories $\phi(t,\underline{\alpha})$ starting at t = 0 with the initial phase at zero.

The squared Euclidean distance between two multi-h signals, N symbols in length, can be written as [2]

$$D^2(N) = \int_o^{NT} [S(t,\underline{\alpha}) - S(t,\underline{\beta})]^2 dt \tag{6}$$

The minimum Euclidean distance is defined as

$$D^2_{min}(N) = \min_{\substack{\underline{\alpha},\underline{\beta} \\ \underline{\alpha} \neq \underline{\beta}}} D^2(N) \tag{7}$$

where $\underline{\alpha}$ and $\underline{\beta}$ are chosen from the set of all possible transmitted sequences. The minimum Euclidean distance is a non-decreasing function of N.

The importance of the minimum Euclidean distance is that it can be used to bound the probability of an error event in the trellis decoding of multi-h. For an AWGN channel application of the union bound gives [3]

$$P_E(\text{error event}) \leq Q\left(\sqrt{D^2_{min}(N)/2N_o}\right) \tag{8}$$

where

$$Q(X) = \int_x^{\infty} \frac{e^{-u^2/2}}{2\pi} du$$

and N_o is the single sided noise power spectral density present at the receiver input. Note that several symbol errors typically occur during each error event.

An upper bound on the minimum Euclidean distance is the free distance between two distinct signals of infinite length [4]:

$$D^2_{free} = \lim_{N\to\infty} D^2_{min} (N) \tag{9}$$

The value of N for which the above limit is attained determines the minimum VA decision depth that is required in the receiver for obtaining the optimum BER performance in an AWGN channel.

For the trellis shown in Figure 1, it can be seen that the first trellis path remerge occurs in 3T seconds. From the structure of the trellis it is clear that for K modulation indices the first remerge will occur in no more than (K + 1)T seconds. It was mentioned earlier that the minimum Euclidean distance is used to bound the probability of error. By increasing the first remerge time, the minimum distance between two signals can be increased, which improves the error performance. The selection of a particular multi-h code also determines the complexity of the VA detector. For modulation indices with common denominator q, a VA detector requires the calculation of 2q branch metrics per symbol. The survivor branches are then used to update the q path histories and metrics stored by the algorithm. The result of this is that as q increases the number of computations required per symbol increases very rapidly.

An additional factor related to receiver complexity is decision depth. As mentioned above, optimum error performance is not achieved if the observation interval used by the decoder for making decisions is not long enough. If the decision depth becomes too long, then the memory requirements are no longer realistic.

III. THE SIMULATION TECHNIQUE

In this section we will describe the approach used to obtain performance results. A block diagram of the multi-h system in terms of simulation parameters is shown in Figure 2. All of the signal processing operations performed up to the multi-h demodulator are in discrete-time complex-baseband form. The transmit filter type, rf 3 dB bandwidth, and center frequency are then varied to obtain various results.

The first operation of the receiver is to process the input signal with a bank of four correlators. This operation is performed symbol by symbol with correlation signals that form an orthonormal set so that the output for each symbol is a four-dimensional signal space vector. Symbol timing can be adjusted by changing the way that the discrete signal samples are grouped together in the correlator. For certain filter functions this is required in order to compensate for signal delay. Carrier phase error has been modeled in the receiver also and results in a rotation of the four-dimensional signal space vectors.

The second operation of the receiver is a VA maximum likelihood sequence estimator. In order to simulate an additive white Gaussian noise channel, independent noise samples are added to each component of each signal space vector before entering the VA. Symbol decisions in the VA are made using the maximum path metric symbol history and the majority vote. Probability of error performance is monitored by the program for both criteria. A tabulation of error run events for sequences of two through ten consecutive symbol errors is also obtained.

IV. RESULTS

In this section the probability of error performance of bandlimited and hardlimited multi-h is considered. The effect of decision depth on bandlimited multi-h is also investigated. Most of the results presented below were obtained using the Monte Carlo simulation program discussed earlier.

Due to the large amount of computer time required to obtain probabilities of error less than 10^{-4}, the lower bound on error events was set at 10. Improved statistical accuracy was achieved at the higher error probabilities where at least 100 error events were obtained.

In all cases a (3/8, 4/8) linear phase code is used with the filtering consisting of a cascade of three first-order Butterworth filters at a lowpass cutoff frequency of 0.55. The cascade cutoff frequency is normalized by the symbol rate. Figure 3 compares the filtered BER performance at decision depths of 9 and 16. Results for the unfiltered case are also shown for comparison.

The reference curve shown in Figure 3 is for coherent MSK with differential encoding. The reason for using this reference is that it is equivalent to VA decoded MSK. With the decision depth fixed at 16, the curves of Figure 4, which show probability of error with and without limiting, were obtained. The power efficiency at $P_E = 10^{-5}$ is summarized for the above results in Table 1. The performance with just bandlimiting can be improved by about 0.35 dB by increasing the decision depth from 9 to 16. Simulation results obtained without filtering showed that the BER performance did not change with decision depth in this region. The filtering has added memory to the system, thus requiring longer decision depths to obtain optimum performance. The degradation due to limiting at a decision depth of 16 is about 0.18 dB. The effect of decision depth on this degradation has not yet been investigated.

An alternate means of obtaining BER performance results is by using the minimum Euclidean distance bounds mentioned earlier. Aulin and Sundberg in [5] have considered CPM using single h-values where the channel contains filtering followed by amplitude limiting. Presently, research is being conducted in this area with application to multi-h. The objective of this study is to compare the minimum Euclidean distance results with those obtained using the Monte Carlo simulation, and to perform additional tradeoff studies.

Preliminary results showing BER degradation at $P_E = 10^{-5}$ versus decision depth for an unfiltered (3/8, 4/8) linear phase code, are shown in Figure 5. The results obtained using the minimum Euclidean distance normalized by the free distance are also shown. The squared free distance of the (3/8, 4/8) REC code, which occurs for a decision of eight, is 2.792 compared with two for MSK, which implies a coding gain of 1.45 dB over MSK. The degradation plots shown above indicate that the two methods predict very similar performance. The variance of the simulation results is more apparent for degradations less than 0.1 dB.

To obtain minimum distance bounds of filtered and limited multi-h codes the concept of minimum Euclidean distance for mismatched signal sets was used [6]. A (2/4, 3/4) REC code was filtered using a cascade of three first-order Butterworth filters with cutoff frequency of 0.5. The unfiltered free distance of this code is 2.788 which occurs at a decision depth of four. Simulation results showing BER degradation at $P_E = 10^{-5}$ are given in Table 2 for filtering and filtering with limiting. The corresponding minimum distance results are also given. It at first appears that the two methods give conflicting results, since the simulation shows reduced degradation with limiting while the minimum distance bounds indicate the opposite is true. It first should be noted that the simulation results are subject to increased measurement variance in the vicinity of $P_E = 10^{-5}$. It also must be realized that the minimum distance results provide asymptotic degradations which may be pessimistic at the SNR values considered.

It was shown earlier that by increasing the decision depth the degradation due to filtering could be reduced. In Table 3 minimum distance degradations are presented for several different decision depths considering both filtering and filtering with limiting. It can be seen that for both cases the degradation increases rapidly when the decision depth drops below four. In going from decision depth four to five the performance of the filtered and filtered/limited cases cross over each other. Note that a negative degradation implies a performance gain over the unfiltered case.

V. CONCLUSIONS

We have described a method for investigating the performance of a multi-h communications system in the presence of practical channel impairments. Computer simulation and minimum distance calculations were used to obtain the results.

It was found that degradations caused by bandlimiting can be minimized by increasing the decision depth. The addition of hardlimiting following the filtering did not change the performance appreciably. The degradation results obtained using the minimum Euclidean are encouraging in that they match the simulation results very closely and can be obtained with considerably less computation time.

ACKNOWLEDGEMENT

This work was performed with partial support from Motorola, Inc., Government Electronics Group.

REFERENCES

[1]. Special Issue on Combined Modulation and Encoding, Guest Editors: J. B. Anderson, J. R. Lesh, IEEE Trans. on Communications, Vol. Com-29, pp. 185-291, March 1981.

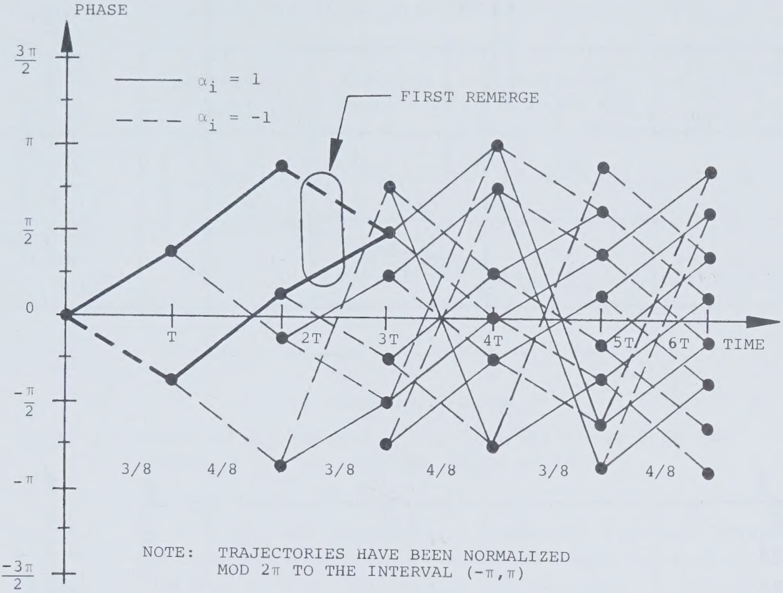
[2]. T. Aulin, C-E. Sundberg, "Continuous Phase Modulation (CPM), Part I: Full Response Signaling", IEEE Trans on Communications, Vol. Com-29, pp. 196-209, March 1981.

[3]. J. M. Wozencraft and I. M. Jacobs, "Principles of Communication Engineering", Wiley 1965.

[4]. C. Hsu, "Multi-h Phase Coding - Its Theory and Design", PhD dissertation, Elec. Eng. Dept., Univ. of Virginia, Charlottesville, VA, 1981.

[5]. T. Aulin and C-E. Sundberg, "Optimum and Sub-Optimum Detectors for Band and Amplitude Limited Continuous Phase Modulation", ICC '83, June 1983, Conf. Rec., pp. A6.1.1-A6.1.6.

[6]. T. Aulin, C-E. Sundberg, and A. Svensson, "Viterbi Detectors with Reduced Complexity for Partial Response Continuous Phase Modulation", NTC '81, Dec. 1981, Conf. Rec., pp. A7.6.1 - A7.6.7.


PHASE
$\frac{3\pi}{2}$
π
$\frac{\pi}{2}$
0
$-\frac{\pi}{2}$
$-\pi$
$-\frac{3\pi}{2}$
$\alpha_i = 1$
$\alpha_i = -1$
FIRST REMERGE
T
2T
3T
4T
5T
6T
TIME
3/8
4/8
3/8
4/8
3/8
4/8

NOTE: TRAJECTORIES HAVE BEEN NORMALIZED MOD 2π TO THE INTERVAL $(-\pi,\pi)$

Figure 1. Phase Trellis for (3/8, 4/8) REC Code

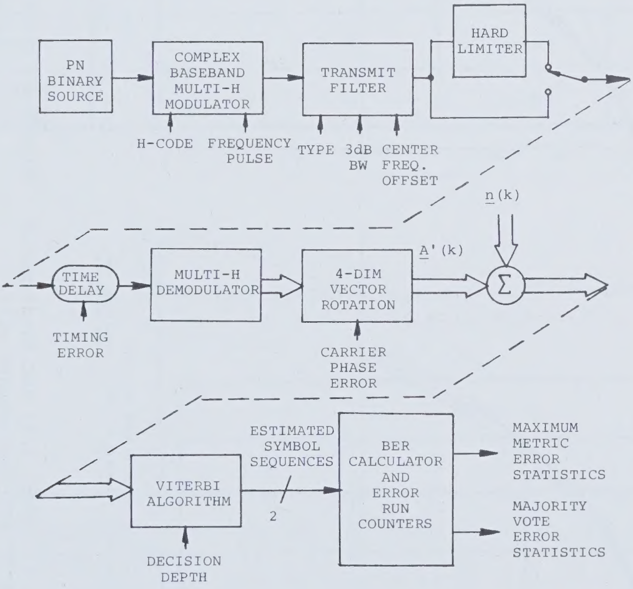

PN BINARY SOURCE
COMPLEX BASEBAND MULTI-H MODULATOR
H-CODE
FREQUENCY PULSE
TRANSMIT FILTER
TYPE
3dB BW
CENTER FREQ. OFFSET
HARD LIMITER
$\underline{n}(k)$
TIME DELAY
TIMING ERROR
MULTI-H DEMODULATOR
4-DIM VECTOR ROTATION
CARRIER PHASE ERROR
$\underline{A}'(k)$
Σ
VITERBI ALGORITHM
DECISION DEPTH
ESTIMATED SYMBOL SEQUENCES
2
BER CALCULATOR AND ERROR RUN COUNTERS
MAXIMUM METRIC ERROR STATISTICS
MAJORITY VOTE ERROR STATISTICS

Figure 2. Simulation System Block Diagram

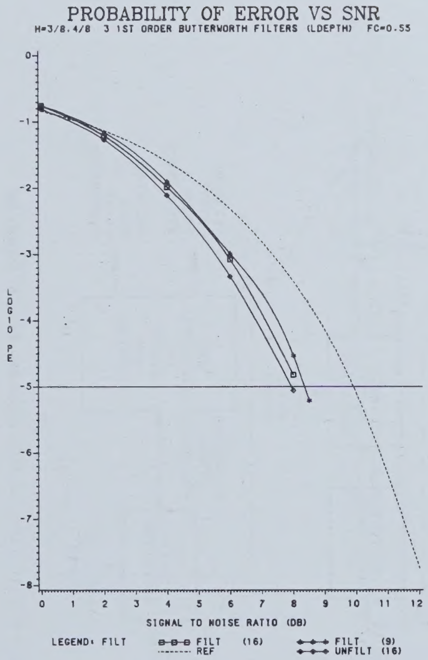


Figure 3. (3/8, 4/8) REC Bandlimited BER Performance with Decision Depth as a Parameter.

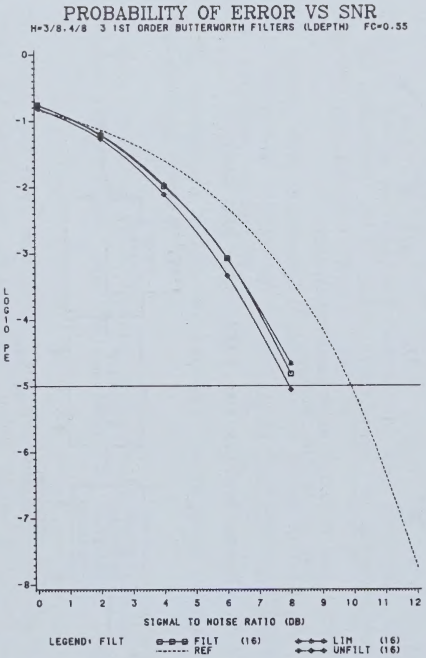


Figure 4. (3/8, 4/8) REC Band-limited and Hardlimited BER Performance.

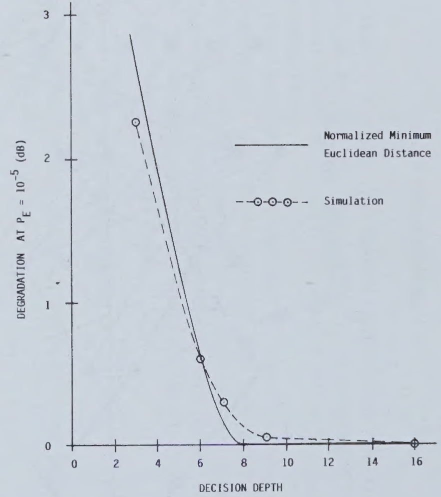


Figure 5. (3/8, 4/8) REC BER Degradation Versus VA Decision Depth.

Table 1. (3/8, 4/8) REC Phase Code BER Performance at PE = 10^{-5}.

	Unfiltered L Depth = 16	Filtered L Depth = 16	Filtered L Depth = 9	Filtered/ Limited L Depth = 16
E_b/N_o in dB at $P_E = 10^{-5}$	7.90	8.19	8.55	8.37
Degradation in dB at $P_E = 10^{-5}$	0	0.29	0.65	0.47

Table 2. Filtered and Limited (2/4, 3/4) REC Phase Code BER Performance with Decision Depth of Four.

	Simulation Results		Minimum Distance Results	
	E_b/N_o in dB at $P_E = 10^{-5}$	Degradation in dB	Normalized D^2_{min}	Degradation in dB
UNFILTERED	7.90	0	2.788	0
FILTERED	8.55	0.60	2.407	0.64
FILTERED/ LIMITED	8.30	0.40	2.316	0.80

Table 3. Minimum Distance Degradation Versus Decision Depth for a Filtered and Limited (2/4, 3/4) REC Code.

	DECISION DEPTH		
	3	4	5
FILTERED	1.93 dB	0.64 dB	0.41 dB
FILTERED/ LIMITED	2.24 dB	0.80 dB	-0.02 dB

Q^2PSK – A NEW SPECTRALLY EFFICIENT MODULATION SCHEME

DEBABRATA SAHA
CICE, The University of Michigan
1520 East Engineering Bldg., Ann Arbor, MI 48109

ABSTRACT

Quadrature-Quadrature phase shift keying (Q^2PSK) is a new spectrally efficient modulation scheme which can be considered as two minimum shift keying type signalling schemes in parallel. It uses two data shaping pulses and two carriers which are pairwise quadrature in phase; the scheme thus utilizes all available signal dimensions in an efficient way to increase the bandwidth efficiency by a factor of two over MSK merely at the expense of fourteen percent increase in the average bit energy.

INTRODUCTION AND PROBLEM BACKGROUND

Spectral congestion due to ever increasing digital transmission always calls for spectrally efficient modulation schemes. Spectrally efficient modulation can be loosely said to be an effort to use power to save bandwidth, much as coding may be said to use bandwidth to save power. In other words the primary objective of a spectrally efficient modulation scheme is to maximize the bandwidth efficiency (b), defined as the ratio of data rate (R_b) to channel bandwidth (W). Like bandwidth, power is also a costly resource in radio transmission. So another objective in designing a high rate data transmission scheme is to reduce the average energy per bit (E_b) for achieving a specified bit error rate (BER). The bit error rate performance of two schemes are usually compared under the assumption of an ideal channel corrupted only by additive white Gaussian noise ($AWGN$). Suppose the two sided power spectral density of noise is $N_0/2$. Then a standard parameter for comparing two modulation schemes is the energy efficiency (e); it is the ratio E_b/N_0 required to achieve a BER of 10^{-5}.

The energy efficiency mostly depends on the signal space geometry. The bandwidth efficiency depends on two factors; firstly the basic waveforms of the data shaping pulses and secondly the utilization of all possible signal dimensions available within the given transmission bandwidth. In data communication, the notion of increasing the rate of transmission by increasing the number of dimensions became prominent when people switched from binary phase shift keying ($BPSK$) to quadrature phase shift keying ($QPSK$). Modulation studies during the last fifteen years proposed several modifications of $QPSK$. Of these, offset quadrature phase shift keying ($OQPSK$) and minimum shift keying (MSK) have gained popularity because of their several attributes. So any new spectrally efficient scheme ought to be tasted in the light of the spectral efficiencies of these two.

$BPSK$ is an antipodal signalling scheme; it uses two opposite phases (0° and 180°) of the carrier to transmit binary $+1$ and -1. Thus $BPSK$ signal space geometry is one dimensional. $QPSK$, on the other hand, can be considered as two $BPSK$ systems in parallel; one with a sine carrier, the other with a cosine carrier of the same frequency. $QPSK$ signal space is thus two dimensional. This increase in dimension without altering the transmission bandwidth increases the bandwidth efficiency by a factor of two. Spectral compactness is further enhanced in MSK by using a cosinusoidal data shaping pulse instead of the rectangular one of $QPSK$. Though MSK and $QPSK$ use different data shaping pulses, their signal space geometries are the same. Both of them use a set of four biorthogonal signals. So the spectral compactness achieved in MSK

over *QPSK* should be distinguished from the compactness achieved in *QPSK* over *BPSK*. In the former compactness comes from the shaping of the data pulse, while in the later it comes from increasing the dimension within the given transmission bandwidth.

To see the possibility of any further increase in dimension without increasing the transmission bandwidth substantially, one has to look into the time-bandwidth product. It is a mathematical truth that the space of signals essentially limited in time to an internal τ and in one sided bandwidth occupancy to W is essentially $2\tau W$-dimensional. Though this bound on dimension is true for the best choice of orthonormal set (prolate spheroidal wave functions [5]), yet it will justify the reasoning behind any search for higher dimensional signal sets to achieve higher bandwidth efficiency. In both *QPSK* and *MSK*, signal duration τ is $2T$, where T is the bit interval in the incoming data stream. Suppose the channel is bandlimited to $1/2T$ on either side of the carrier, i.e. one sided bandwidth occupancy is $W = 1/T$. With such a bandlimited channel a *QPSK* system will be able to transmit only ninety percent of its total power while an *MSK* system transmits ninety seven percent. The number of dimensions available within this bandwidth $W = \frac{1}{T}$ is $2\tau W = 4$. It is surprising that only two of them are utilized in *QPSK* and *MSK*. The remaining two are yet left to be played around with. So one can intuitively think of a modulation scheme with a bandwidth efficiency as much as twice that of *QPSK* or *MSK*. Since prolate spheriodal wave functions are not easy to implement, expectation of hundred percent increase in bandwidth efficiency may be too much from a practical view point. Yet the extra two dimensions give some room for improving the bandwidth efficiency by increasing the dimensionality of the signal set at the cost of some extra bandwidth, if necessary. The object of this paper is to demonstrate such an example.

Next we briefly review some existing modulation techniques such as *QPSK*, *OQPSK* and *MSK*. Then we propose a new modulation technique named Quadrature - Quadrature Phase Shift Keying or Q^2PSK. We analyze the energy and bandwidth efficiency of Q^2PSK and compare with those of *MSK*.

BRIEF REVIEW OF QPSK, OQPSK and MSK

A block diagram of the *QPSK* modulation scheme is shown in Fig.1. The input binary data stream $\{a_k\}$ arriving at the rate of $1/T$ is demultiplexed into two streams $a_1(t)$ and $a_2(t)$. The duration of each bit in the demultiplexed streams is twice the duration of the incoming bit. Streams $a_1(t)$ and $a_2(t)$ are multiplied by sine and cosine carriers and summed up to form the *QPSK* signal $s_{qpsk}(t)$.

$$s_{qpsk}(t) = \frac{1}{\sqrt{2}}\, a_1(t)\, Cos\,(2\pi f_c t + \frac{\pi}{4}) + \frac{1}{\sqrt{2}}\, a_2(t)\, Sin\,(2\pi f_c t + \frac{\pi}{4})$$
$$= Cos\,(2\pi f_c t + \varphi(t)) \qquad (1)$$

where $\varphi(t)$, depending on $a_1(t)$ and $a_2(t)$, is one of 0°, ±90° and 180° Thus carrier phase during any $2T$ interval is one of the four phases. In the next $2T$ interval, if neither of the two bit streams changes sign, the carrier phase remains the same. If one of them changes sign, a phase shift of ±90° occurs. A change of sign in both streams causes a phase shift of 180°. Rapid changes in the carrier phase has serious deteriorating effects on the signal and the adjacent channel when it undergoes bandlimiting and hardlimiting operations. Those deteriorating effects are partially eliminated in offset quadrature phase shift keying (*OQPSK*) where the two bit streams are not allowed to change their sign simultaneously, thus avoiding the possibility of 180° phase shift. This is accomplished by skewing or delaying the bit stream $a_2(t)$ by an amount of time

T as illustrated in Fig.2. By pulse shaping, further elimination of changes in the carrier phase is possible. In fact, it is totally eliminated in minimum shift keying (*MSK*) where a cosinusoidal data shaping pulse is used in place of the rectangular one of *OQPSK*, as illustrated in Fig.2. This pulse shaping in *MSK* also brings some spectral compactness over *OQPSK*. The *MSK* signal can be written as

$$s_{msk}(t) = Cos\,[2\pi(f_c + \frac{b(t)}{4T})t + \varphi(t)] \tag{2}$$

where $b(t) = -a_1(t)a_2(t)$ and $\varphi(t) = 0$ are π according to $a_1 = +1$ or -1. Thus *MSK* signals can have one of two instantaneous frequencies, $f_c \pm \frac{1}{4T}$. The spacing between the two frequencies is $\frac{1}{2T}$. This is the minimum spacing with which two *FSK* signals can be orthogonal; hence the name minimum shift keying (*MSK*).

QUADRATURE - QUADRATURE PHASE SHIFT KEYING

Let us consider the following signal set.

$$s_1(t) = \cos(\pi t/2T)\cos 2\pi f_c t \quad , |t| \leq T \tag{3a}$$

$$s_2(t) = \sin(\pi t/2T)\cos 2\pi f_c t \quad , |t| \leq T \tag{3b}$$

$$s_3(t) = \cos(\pi t/2T)\sin 2\pi f_c t \quad , |t| \leq T \tag{3c}$$

$$s_4(t) = \sin(\pi t/2T)\sin 2\pi f_c t \quad , |t| \leq T \tag{3d}$$

$$s_i(t) = 0 \quad , i = 1,2,3,4\, , |t| > T \tag{3e}$$

We write,

$$p_1(t) = \cos(\pi t/2T) \text{ and} \tag{4a}$$

$$p_2(t) = \sin(\pi t/2T) \tag{4b}$$

Later $p_1(t)$ and $p_2(t)$, which are quadrature in phase, will be identified as data shaping pulses, and sine and cosine functions of frequency f_c as carriers. It is to be noted that $\{s_i(t)\}$ is a set of four equal-energy orthogonal signals under the restriction

$$f_c = n/4T \quad , n = \text{integer} \geq 2 \tag{5}$$

Also the othogonality remains invariant under the translation of the time origin by multiples of $2T$, which is the duration of each signal. In other words, if the definition of $s_i(t)$ in (3) be extended for all t, then one will get orthogonality over every interval of $2T$ centered around $t = 2mT$, m being an integer.

The orthogonality of $\{s_i(t-2mT)\}$ suggests a modulation scheme, a schematic diagram of which is shown in Fig.3. Data from an *IID* binary (± 1) source at a rate $2/T$ is demultiplexed into four streams $\{a_i(t)\}$; duration of each data pulse (rectangular shaped) in the demultiplexed streams being $2T$. Each data stream $a_i(t)$ is multiplied by the output $s_i(t)$ of a signal generator which continuously emits $s_i(t)$, defined over all t. The multiplied signals are summed up to form the modulated signal $s(t)$.

At the receiver, suppose four identical coherent generators are available. Then one can make observations over intervals of length $2T$ and use the orthogonality of $\{s_i(t-2mT)\}$ to separate out the data streams. A correlation receiver can perform this process of demodulation in an optimum sense of minimum probability of error in the presence of white Gaussian noise.

The bit streams $a_1(t)$ to $a_4(t)$ all are synchronously aligned at intervals of $2T$. Like *QPSK* this will cause abrupt changes in the carrier phases. These undesirable abrupt changes can be reduced by staggering or delaying the bit streams pairwise. This is accomplished by giving $a_3(t)$ and $a_4(t)$ a time offset T or equivalently an offset T to the data shaping pulses associated with $a_3(t)$ and $a_4(t)$ as illustrated in Fig.4. It is to be noted this offset will not cause any ambiguity in separating out the signals at the receiver. The dimensionality of the signal set used in this scheme is four; two of them come from the orthogonality of the carriers, the remaining two from the orthogonality of the data shaping pulses $p_1(t)$ and $p_2(t)$. In other words, two carriers and two data shaping pulses are pairwise quadrature in phase. Hence the modulation scheme is named Quadrature - Quadrature Phase Shift Keying (Q^2PSK).

Looking at the schematic diagram in Fig.3 and the Q^2PSK-pulse trains in Fig.4, one can represent the Q^2PSK signal as

$$s_{q^2psk}(t) = \cos[2\pi(f_c + b_{13}(t)/4T)t + \varphi_{13}(t)] + \sin[2\pi(f_c + b_{24}(t)/4T)t + \varphi_{24}(t)] \tag{6}$$

where

$$b_{13}(t) = -a_1(t)\, a_3(t) \tag{7a}$$

$$\varphi_{13}(t) = 0 \text{ or } \pi \text{ according to } a_1(t) = +1 \text{ or } -1 \tag{7b}$$

and

$$b_{24}(t) = +a_2(t)\, a_4(t) \tag{8a}$$

$$\varphi_{24}(t) = 0 \text{ or } \pi \text{ according to } a_4(t) = +1 \text{ or } -1 \tag{8b}$$

Thus at any instant the Q^2PSK signal can be analyzed as consisting of two signals; one is cosinusoidal with frequency either of $(f_c \pm 1/4T)$, the other is sinusoidal with frequency either of $(f_c \pm 1/4T)$. The separation between the two frequencies associated with either of the two signals is $1/2T$; this is the minimum spacing that one needs for coherent orthogonality of two *FSK* signals as in *MSK*. Also a comparison with (2) shows that the cosinusoidal part of Q^2PSK signal in (6) exactly represents an *MSK* signal. Therefore the Q^2PSK signalling scheme can be thought as consisting of two minimum shift keying type signalling (*MSKTS*) schemes, one in quadrature with the other. Since the two *MSKTS* are in quadrature, one can intuitively think that overall energy efficiency will be the same as that of conventional *MSK* with cosinusoidal shape of data pulse. In the next section we verify this intuition.

ENERGY EFFICIENCY

Usually the *BER* performance is evaluated under the assumption of *AWGN* and an optimum receiver. A standard parameter for measuring the performance is the energy efficiency; it is the ratio E_b/N_0 required to achieve a *BER* $P_b(E) = 10^{-5}$.

The signal set $\{s_i(t)\}$ used in Q^2PSK is of dimension $N = 4$. Each $s_i(t)$ represents one of four co-ordinate axes. With respect to this set of axes, a Q^2PSK signal can be represented as

$$s(t) = [a_1(t), a_2(t), a_3(t), a_4(t)] \tag{9}$$

where the coordinates $a_i(t)$'s can have only one of two values ± 1. The number of signals in the Q^2PSK signal set is 2^4. The signals are equiprobable and of equal energy, say E_s. Also it is easy to check that they represent the vertices of a hyper cube of dimension four; the centre of the cube being at the origin of the coordinate axes. For this signal space geometry, the signal error probability is given by

$$P_s(E) = 1 - (1-p)^4 \tag{10}$$

where

$$p = Q(\sqrt{E_s/2N_0}) \tag{11}$$

and

$$Q(v) = \sqrt{1/2\pi}\int_v^\infty \exp(-x^2/2)dx \tag{12}$$

Knowing signal error probability, one has upper and lower bound on bit error probability given by

$$\frac{1}{4}\ P_s(E) \le P_b(E) \le P_s(E) \tag{13}$$

However an exact calculation of $P_b(E)$ is of considerable interest for comparing two modulation schemes. Each of sixteen possible Q^2PSK signals represents a combination of four bits. If $P_{bi}(E)$ is the probability of error in the i^{th} bit position, then

$$P_b(E) = \frac{1}{4}\sum_{i=1}^{4} P_{bi}(E) = P_{b1}(E) \tag{14}$$

where the last equality comes from the equality of $P_{bi}(E)$ for all i because of the symmetry in signal space geometry. To calculate $P_{b1}(E)$ let us divide the signals into two groups: $\{(+1,a_2,a_3,a_4)\}$ and its image partner $\{(-1,a_2,a_3,a_4)\}$, where a_i 's can take values of either ± 1. Then one can imagine a hyper plane which separates the two groups on its either side and is equidistant from each group. The distance of any signal in either group from this hyper plane is $\frac{d}{2} = \sqrt{E_s/N} = \sqrt{E_b}$. Thus the signals with $+1$ in the first bit position are on one side of this plane at a distance $\sqrt{E_b}$ while the signals with -1 in the first bit position are on the other side at the same distance. So an error in the first bit position occurs only when the noise component $n(t)$ associated with this bit position drives a signal down to the other side of the hyper plane. The probability of such an incident is

$$P_{b1}(E) = \int_{\sqrt{E_b}}^{\infty} p_n(x)dx = Q\left[\sqrt{\frac{2E_b}{N_0}}\right] \tag{15}$$

where $p_n(x)$ is the probability density function of Gaussian noise with two sided spectral density $\frac{N_0}{2}$. Hence the overall bit error probability is

$$P_b(E) = P_{b1}(E) = Q\left[\sqrt{\frac{2E_b}{N_0}}\right] \tag{16}$$

The energy efficiency of the Q^2PSK scheme is therefore $9.6db$. The derivation of (16) does not assume any particular dimension N of the hyper cube; it is valid for any N. $BPSK$ use two antipodal signals which can be considered as the vertices of a hyper cube of dimension one. Similarly $QPSK$ and MSK, which use a set of four biorthogonal signals, can be considered as using the vertices of a hyper cube of dimension two. So all of $BPSK$, $QPSK$, $OQPSK$, MSK and Q^2PSK belong to the same class of signalling schemes which use vertices of some hyper cube, and each of them has an energy efficiency $9.6db$.

SPECTRAL DENSITY AND BANDWIDTH EFFICIENCY

One can represent a Q^2PSK signal as

$$s_{q^2psk}(t) = \sum_{i=1}^{2} \frac{1}{\sqrt{T}}\, a_i(t)s_i(t) + \sum_{i=3}^{4} \frac{1}{\sqrt{T}}\, a_i(t)s_i(t-T) \tag{17}$$

where the additional $1/\sqrt{T}$ is just a normalizing factor to make $\frac{1}{\sqrt{T}} Cos\left[\frac{\pi t}{2T}\right]$ and $\frac{1}{\sqrt{T}} Sin\left[\frac{\pi t}{2T}\right]$ unit energy pulses. Each $a_i(t)$ can be either $+1$ or -1 with probability one half. So the Q^2PSK signal can be one of sixteen possible equiprobable waveforms. Let us represent these waveforms by $m_i(t)$, i varying from 1 to 16. Probability of occurrence of $m_i(t)$ is $p_i = 1/16$ for all i. The signal set $\{m_i(t)\}$ has the following characteristics:

(i) for each waveform $m_i(t)$, there is also a waveform $-m_i(t)$

(ii) the stationary probability of $m_i(t)$, and $-m_i(t)$ are equal

(iii) the transitional probabilities between any two waveforms are the same.

Such a signalling source is said to be negative equally probable (NEP); the overall spectral density is given by [7]

$$S_{q^2psk}(f) = \sum_{i=1}^{16} p_i\, |M_i(f)|^2 \tag{18}$$

where $M_i(f)$ is the Fourier transform of $m_i(t)$. One can reasonably assume the carrier frequency $f_c >> \frac{1}{T}$; then for $f > 0$, each $M_i(f)$ is one of the following sixteen possible combinations.

$$\frac{1}{2}\left[\pm P_1(f-f_c) \pm P_2(f-f_c) \pm jP_1(f-f_c)e^{-jWT} \pm jP_2(f-f_c)e^{-jWT}\right]$$

where $P_1(f)$ and $P_2(f)$ are the Fourier transforms of data shaping pulses $p_1(t)$ and $p_2(t)$ given by

$$p_1(t) = \frac{1}{\sqrt{T}}\, Cos\, \frac{\pi t}{2T}\, \Bigg|_{-T}^{T} \tag{19}$$

$$p_2(t) = \frac{1}{\sqrt{T}}\, Sin\, \frac{\pi t}{2T}\, \Bigg|_{-T}^{T} \tag{20}$$

Substituting the $M_i(f)$'s into (18) and noticing that all cross terms are cancelled out, one can write

$$S_{q^2psk}(f) = \frac{1}{2}\left[|P_1(f-f_c)|^2 + |P_2(f-f_c)|^2\right] \tag{21}$$

Equivalent baseband version of spectral density is

$$S^b_{q^2psk}(f) = \frac{1}{2}\left[|P_1(f)|^2 + |P_2(f)|^2\right] \tag{22}$$

The Fourier transforms of the data shaping pulses are

$$P_1(f) = \frac{4\sqrt{T}}{\pi}\left[\frac{\cos 2\pi f T}{1-16f^2T^2}\right] \tag{23}$$

$$P_2(f) = -\frac{j16\sqrt{T}}{\pi}\left[\frac{fT\cos 2\pi f T}{1-16f^2T^2}\right] \tag{24}$$

Substituting (23) and (24) into (22), the spectral density is found to be

$$\frac{1}{T}\, S^b_{q^2psk}(f) = \left[\frac{8}{\pi^2}\right]\left[1+16f^2T^2\right]\left[\frac{Cos\, 2\pi f\, T}{1-16f^2T^2}\right]^2 \tag{25}$$

Similar derivations give spectral densities of MSK and $OQPSK$ signalling schemes as

$$\frac{1}{T}\, S^b_{msk}(f) = \left[\frac{16}{\pi^2}\right]\left[\frac{Cos\, 2\pi f\, T}{1-16f^2T^2}\right]^2 \tag{26}$$

$$\frac{1}{T}\, S^b_{oqpsk}(f) = 2\left[\frac{Sin\, 2\pi f\, T}{2\pi f\, T}\right]^2 \tag{27}$$

where in all cases,

$$\int_{-\infty}^{\infty} S^b(f)df = 1 \tag{28}$$

Spectral densities of $OQPSK$, MSK and Q^2PSK are sketched in Fig.5. Q^2PSK uses two different kinds of data pulses; one is $p_1(t)$ having a cosinusoidal shape as in MSK, the other is $p_2(t)$ having a sinusoidal shape. The shape of $p_1(t)$ is smoother than $p_2(t)$ in the sense that the later has jumps at $t = \pm T$; as a result, for large f, the spectral fall-off associated with $p_2(t)$ is proportional to f^{-2} while that with $p_1(t)$ is as f^{-4}. The faster fall-off associated with cosinusoidal shape causes lower side lobes in MSK; side lobes in $OQPSK$ and Q^2PSK are of the same order in magnitude but relatively higher than those of MSK.

A measure of spectral compactness is the percent of total power captured in a specified bandwidth. This is plotted in Fig.6. For small bandwidth, the percent power captured in Q^2PSK is smaller than that in $OQPSK$ and MSK. Beyond a bandwidth of $1.2/T$, the asymptotic behaviour of $QPSK$ and Q^2PSK become almost identical because of their same type of spectral fall-off as f^{-2}. MSK captures 99.1% of total spectral power in a bandwidth of $W = 1.2/T$. With the same bandwidth power captured in $QPSK$ and Q^2PSK are 90.6 and 91.13 percent respectively. Thus MSK seems to be more spectrally compact than Q^2PSK; yet bandwidth efficiency of Q^2PSK is higher because its data transmission rate is twice that of MSK. An exact calculation of bandwidth efficiency depends on the definition of bandwidth and the effect of bandlimiting on signal components.

Q^2PSK Vs MSK

Consider an existing MSK scheme which allows a bandwidth of $1.2/T$ so that almost the entire spectrum (99.1% power) is available at the receiver. Suppose the MSK-modulator is replaced by Q^2PSK-modulator and the modulator output is still bandlimited to $1.2/T$. Our object is to compare the energy and bandwidth efficiencies of the bandlimited Q^2PSK with the existing MSK scheme.

A bandwidth of $1.2/T$ allows transmission of 91.13% of total Q^2PSK spectral power. Thus there is a loss of some spectral components; this loss causes spread of the data pulses which in turn causes intersymbol interference (ISI). The effect of this ISI can be equivalently considered as changing the signal space geometry. With respect to the Q^2PSK basis-set $\{s_i(t)\}$, the signal space geometry no longer remains a hypercube; one can show that the geometry turns into rectangular hyper parallelopiped of dimension four. It is interesting and important to note that the lengths of the parallelopiped sides are dependent on the data sequence. Essentially there are sixteen different data dependent situations each of which has equal probability of occurrence, but different parallelopiped for signal space geometry. One can show through some analysis and computations that with this set of signal space geometries for bandlimited Q^2PSK scheme the E_b/N_0 requirement for a bit error rate of 10^{-5} is 10.17 *db* while that for *MSK* is 9.6 *db* . Thus Q^2PSK achieves twice the bandwidth efficiency of *MSK* only at the expense of 14% increase in the average bit energy.

ACKNOWLEDGEMENT

The author is sincerely grateful to William L. Root for his advice and assistance in compiling this paper.

REFERENCES

(1) Claude E. Shanon, "Communication in the presence of noise," Proc. IRE 37, No.1, pp.10-21, January 1949.

(2) Subbarayan Pasupathy, "Minimum shift keying: a spectrally efficient modulation," IEEE Communications Magazine, pp.14-22, July 1979.

(3) J. G. Smith, "Spectrally efficient modulation," Proc. IEEE Int. Conf. Commun. (ICC'77), June 1977, pp.3.1-37 - 3.1-41.

(4) H. J. Landau and H. O. Pollak, "Prolate spheroidal wave functions, Fourier analysis and uncertainty-III: The dimensions of the space of essentially time and band limited signals," BSTJ., 41, pp.1295-1336, July 1962.

(5) J. M. Wozencraft and I. M. Jacobs, Principle of Communication Engineering, John Wiley and Sons, 1965.

(6) W. C. Lindsey and M. K. Simon, Telecommunication Systems Engineering, Prentice-Hall, Englewood Cliffs, N.J., 1973.

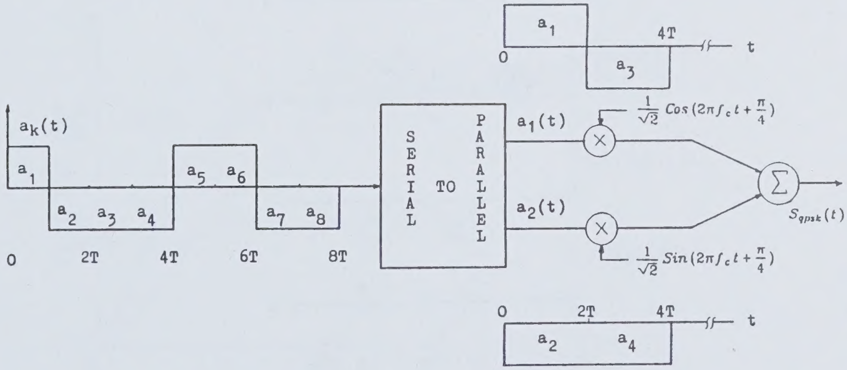

a_1
4T
t
0
a_3
$\frac{1}{\sqrt{2}} Cos(2\pi f_c t + \frac{\pi}{4})$
$a_k(t)$
a_1
a_5 a_6
a_2 a_3 a_4
a_7 a_8
0
2T
4T
6T
8T
SERIAL TO PARALLEL
$a_1(t)$
$a_2(t)$
Σ
$S_{qpsk}(t)$
$\frac{1}{\sqrt{2}} Sin(2\pi f_c t + \frac{\pi}{4})$
0
2T
4T
t
a_2
a_4

Fig.1. Quadrature Phase Shift Keying modulator.

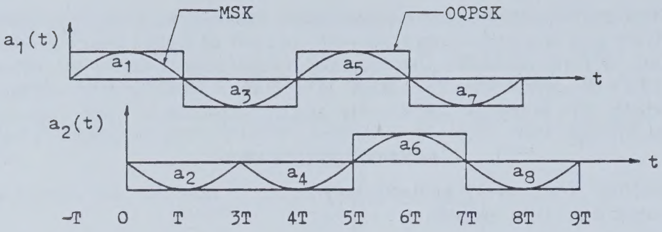


Fig.2. Staggering of data pulses in OQPSK and MSK for the data stream $a_k(t)$ in Fig.1.

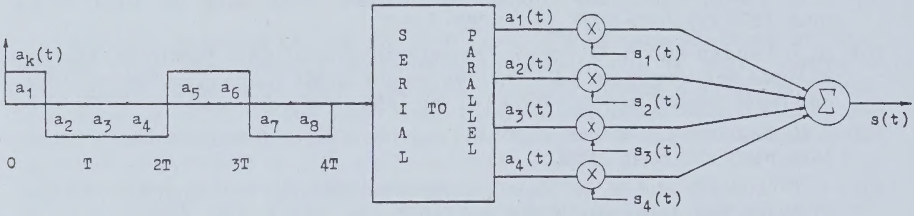

$a_k(t)$
a_1
a_2 a_3 a_4
a_5 a_6
a_7 a_8
0
T
2T
3T
4T
SERIAL TO PARALLEL
$a_1(t)$
$a_2(t)$
$a_3(t)$
$a_4(t)$
$s_1(t)$
$s_2(t)$
$s_3(t)$
$s_4(t)$
$\sum$
s(t)

Fig.3. Quadrature-Quadrature Phase Shift Keying(Q^2PSK)modulator.

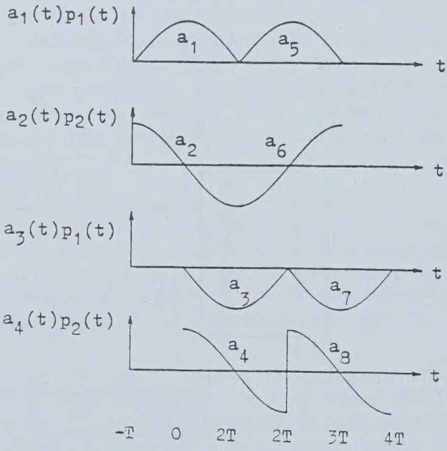


Fig.4. Staggering of data pulses in Q^2PSK for the data stream $a_k(t)$ in Fig.3.

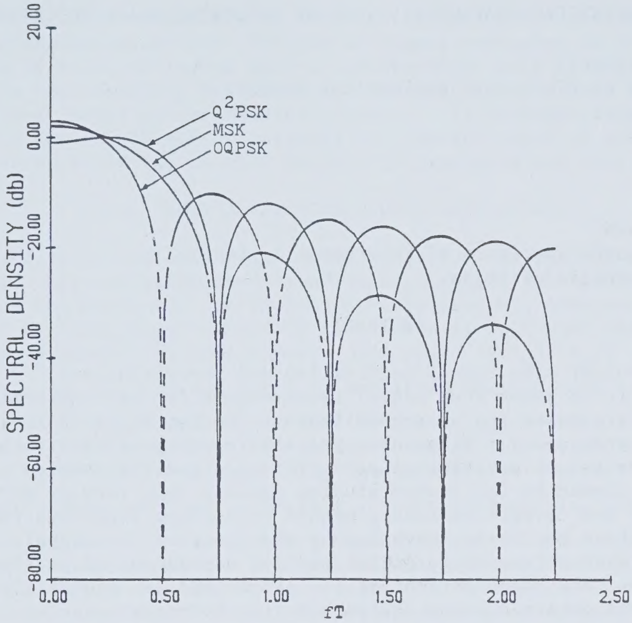


Fig.5. Spectral densities of OQPSK,MSK,and Q^2PSK.

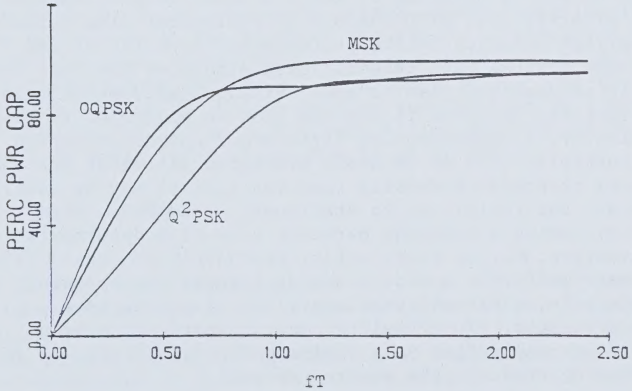


Fig.6. Percent power captured as function of bandwidth.

FIRST ORDER STATISTICAL CHARACTERISATION OF CYCLOSTATIONARY PROCESSES

JOHN J O'REILLY
Department of Electrical Engineering Science
University of Essex
COLCHESTER
CO4 3SQ
England

WANG JUN JUN
Wuhan Research Institute of Telecommunications
Peoples Republic of China

ABSTRACT

Limits to equivalent time signal acquisition and processing are considered and it is shown that useful measurements for both deterministic and stochastic processes can be accomplished. The latter are not required to be stationary - widesense cyclostationarity is sufficient. This is an important observation since very many signals/processes encountered in communication system studies exhibit this form of periodic nonstationarity and certain necessary signal processing functions rely for their operation on the cyclostationary character of the signal. This is illustrated with reference to jitter and isi assessment of baseband digital signals. Histogram estimators for jitter and isi probability density functions obtained using equivalent time techniques are considered and the implications for measurement precision of certain impairments inherent to this form of signal acquisition are discussed. In particular, the confluence of acquisition time jitter and signal averaging is shown to result in a form of aperture distortion with asymptotic properties related to the first order jitter statistics via fractional order calculus.

1. INTRODUCTION

The acquisition and post-processing of certain deterministic signals is now frequently carried out with the aid of equivalent time methods [e.g. 1] and analyses relating to the performance capabilities and limitations of such systems have recently been presented for this case [2]. However, it is possible also to make certain measurements relating to stochastic processes and this is the new problem addressed in this paper. In particular, a comprehensive first order characterisation is found to be possible since an accurate histogram estimator for the underlying process probability density function (p.d.f) may be obtained. Furthermore, we are not restricted to stationary processes. We illustrate this by taking a somewhat perverse view of a deterministic signal as a degenerate, single realization, stochastic process. This allows us to treate uniformly a wide class of signals encompassing periodic deterministic, widesense stationary and also widesense *cyclostationary* processes. In actuality, measurements are made on segments of a single realization so a *cycloergodic hypothesis* [3] is involved in extending these to the entire process.

To provide a focus for the discussion we consider specifically certain measurements related to digital communications. In particular, we present illustrative results for timing jitter and intersymbol interference (i.s.i) measurements on binary baseband systems and analyse the influence of timing jitter on system pulse/frequency response characterisation. The use of signal averaging in the latter instance is found to induce aperture distortion, with frequency domain asymptote corresponding to some (possibly fractional order) derivative of the first order jitter density function. Illustrative results are presented for likely jitter patterns and the influence of temporal quantization noise [4] in this context is evaluated explicitly.

2. EQUIVALENT-TIME SIGNAL ACQUISITION

Consider first a periodic signal $y(t) = y(t+T)$. We wish to acquire a sampled representation corresponding to a single realization of one period of the waveform. In equivalent-time systems, however, this is achieved by sampling at arbitrary time instants both $y(t)$ and a related timebase waveform. We thus acquire data pairs $(y(t_k), t_k)$. The first element $y(t_k)$ is subjected to amplitude quantization in the usual way so that it can be coded and stored at some appropriate location in an n-bit wide data memory. The memory location is determined by reducing t_k module T and quantizing the residue; we denote the resultant equivalent-time value by $\tilde{t}_n$. As a consequence, data values which are adjacent in memory are generally derived from disjoint realization segments of the periodic signal and even the *equivalent-time* at which data are acquired is uncertain to within one half of the time axis quantel step size. Hence any additive noise is largely decorrelated as illustrated by figure 1, provided Min $\{t_{n+1}-t_n\} \gg \tau_c$, the correlation interval of the noise, and this has implications for subsequent post-processing [2]. Also, the time quantization error introduces an effective noise component into the acquired data even if the input signal is noise free [4]. While being broadly similar in character this differs somewhat in detail from the familiar amplitude quantization noise and has implications for both the design and use of equivalent-time measurement systems. Analysis shows that compared with real-time systems equivalent-time systems can, for a given sample density, offer improved or degraded performance on subsequent optimal post processing if coloured additive noise impairs the signal. The result is sensitive to the relationship between the signal and noise spectra and signal averaging improves both systems similarly. We assume in this that the signal and noise are uncorrelated. So far as time quantization is concerned this is a clear degradation unique to equivalent-time schemes and we find that the signal to noise ratio attainable is given by:

$$SNR = \frac{3M}{(\pi\tau\beta)^2} \qquad \ldots (1)$$

where M is the signal averaging factor, τ the effective sample spacing in equivalent-time, and β is the rms bandwidth of the signal. It is interesting to compare this with the familiar SNR limit imposed by amplitude quantization [5]. If the signal impairment from each source is to be comparable we deduce that the number of bits J required for time quantization differs from the number of bits K required for amplitude quantization by a factor:

$$J-K = \log_2 \{\pi\ N\ (\beta P)\ \frac{\overline{y^2}}{M}\} \qquad \ldots (2)$$

	gaussian	uniform	bi-phase	sinusoidal
p.d.f. $p_{\varepsilon}(t)$	$\frac{\exp(-t^2/2\sigma^2)}{\sqrt{2\pi}\ \ \sigma}$	$\frac{1}{2\varepsilon}$; $\lvert t\rvert<\varepsilon$ 0 elsewhere	$\frac{1}{2}\ \delta(t\pm\varepsilon)$	$\frac{1}{\pi\sqrt{\varepsilon^2-t^2}}$; $\lvert t\rvert<\varepsilon$ 0 elsewhere
variance $\sigma_j^{\ 2}$	σ^2	$\frac{\varepsilon^2}{3}$	ε^2	$\frac{\varepsilon^2}{2}$
effective filter, $P_{\varepsilon}(f)$	$\exp\{-2(\pi\sigma f)^2\}$	$\frac{\sin(2\pi\varepsilon f)}{2\pi\varepsilon f}$	$\cos(2\pi\varepsilon f)$	$J_o(2\pi\varepsilon f)$
asymptotic behaviour	undefined	$f^{-1} \equiv -6$ dB/octave	$f^0 \equiv 0$ dB/octave	$f^{\frac{1}{2}} \equiv -3$ dB/octave

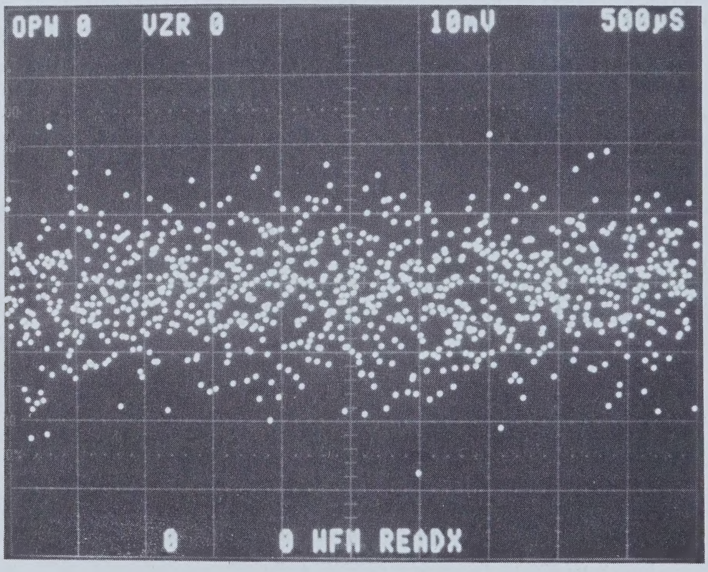


Figure 1 Decorrelated Equivalent-time realisation of a noise waveform

Here N denotes the number of waveform periods acquired, (βP) is the product of RMS bandwidth and period of the signal and $\overline{y^2}$ is the mean square value of the signal normalized for maximum excitation of the amplitude quantizer. Notice that in the absence of signal averaging time quantization error is generally more problematic than amplitude quantization and markedly so if the high frequency content of the signal is significant $(\beta P >> 1)$. Hence, in allocating a fixed number of bits of storage to a system we do better if we quantize more finely the time axis than the amplitude axis. Of course, we can reduce the time quantization noise by signal averaging but, as we shall see shortly, this introduces aperture distortion. We are thus involved in a trade-off between a stochastic and an essentially deterministic impairment.

3. EQUIVALENT-TIME MEASUREMENTS ON STOCHASTIC PROCESSES

Let us now consider a widesense stationary stochastic process $Y(t)$ with $y(t)$ an arbitrarily selected realization. Invoking ergodicity arguments we can sample $y(t)$ at times t_i and take the $y(t_i)$ as representative samples of $Y(t)$. Hence, trivially, we can obtain a histogram estimator for the probability density function of $Y(t)$ from almost any $y(t)$. In digital communications, however, signal processes of interest are rarely stationary even in the wide sense. For example if we assume a wide sense stationary data sequence $\{a_n\}$ mapped onto the periodic pulse sequence $\sum_n p(t-nT)$ using pulse amplitude modulation the resultant $s(t) = \sum_n a_n p(t-nT)$ is in general *cyclostationary* rather than *stationary*. Random sampling of a "representative" realization gives rise to statistics corresponding to the associated phase randomised process [6]. However, the statistics of a cyclostationary process vary periodically with time - indeed, such processes are often referred to as *periodic* non-stationary. If for an equivalent-time system the timebase can be synchronized to this underlying period then we can operate as for the deterministic signal, placing sample points $y(t_k)$ in appropriate storage locations determined by reducing t_k modulo T and quantising the residue. Time varying statistics for the cyclostationary process, including the time varying first-order p.d.f, are then readily obtained by acquiring a sequence of 'equivalent' waveforms. The latter will generally not correspond to possible realisation segments of the process $Y(t)$ due to the decorrelating influence of the equivalent-time sampling process but first-order statistics are preserved. This time a cycloergodic hypothesis is involved [3]. We observe in passing that with this perpespective a deterministic periodic signal corresponds to a zero variance, time-varying mean, cycloergodic process.

4. SOME ILLUSTRATIVE MEASUREMENT SYSTEMS

Consider a baseband digital transmission system with signalling element $p(t)$ and data sequence $\{a_n\}$. The waveform process then has the form:

$$s(t) = \sum_n a_n p(t-nT+\delta_n) \qquad \ldots (3)$$

where $\{\delta_n\}$ represents random jitter in the timing of the signal elements. If we sample $s(t)$ at times nT we obtain $\{\hat{a}_n\}$. If now $\delta_n \to 0$ and

$p(nT) = 0 \forall n \neq 0$ then $\{\hat{a}_n\} = \{a_n\}$ and we have recovered the data. If, though, $p(nT) \neq 0 \forall n \neq 0$ the system is said to suffer from intersymbol interference (i.s.i) and this manifests itself as vertical closure in the system eye pattern leading to a degradation in error probability (P_e) in the presence of additive noise. The precise relationship between i.s.i and P_e depends on the form of i.s.i p.d.f; a histogram estimator for the latter is readily obtained using equivalent time acquisition on the system eye pattern. This requires only synchronization to the system signalling rate or data clock. The i.s.i distribution is obtained by examining the values entering the waveform store location corresponding to the sample time or time of maximum vertical eye opening. An illustrative binary eye pattern is shown in figure 2 from which the cyclostationary character of the signal process is apparent.

Similarly, if we wish to measure the jitter we simply display the eye pattern and observe the times at which the transitions cross a fixed "decision" threshold. It is often convenient to employ a window surrounding this threshold thereby trading resolution for reduced measurement time and there is benefit, too, to be obtained by adopting a suitable window weighting function [7]. A measured jitter distribution is shown in figure 3 for illustration.

Let us now consider measuring the system response in the presence of jitter. The input signal has the form:

$$y(t) = \sum_n p(t - n\tau_s + \delta_n) \qquad \dots (4)$$

where $\tau_s >> T$ is the period of the test signal and the jitter δ_n has p.d.f. $f_j(t)$. If we employ signal averaging then the stochastic impairment due to δ_n will be asymptotically eliminated but this has the effect of convolving $p(t)$ with the time reverse of $f_j(t)$. In the frequency domain this manifests itself as aperture distortion: the system frequency response $P(f)$ is weighted by the characteristic function of the jitter. Results for some illustrative jitter statistics are presented in table 1. Note that the effective filtering depends only on the first order characteristics of the jitter. Correlation can influence convergence but does not generally do so in view of the decorrelating properties of the equivalent-time process. Note also that the asymptotic behaviour of the filter is readily obtained, via Fourier theory, by considering the differentiation order of the jitter p.d.f required to induce an impulsive character to the resultant. This may, of course, involve a fractional order differential as exemplified by the case of sinusoidal jitter giving rise to a Bessel function aperture (zero-order, first kind) with $f^{-\frac{1}{2}}$ asymptote.

So far as measurements on *stationary* processes are concerned time jitter is of no importance since $Y(t) = Y(t+\varepsilon) \forall \varepsilon$. Not so, though, for *cyclostationary* processes; time-jitter imposes an effective filtering on the cyclical variation of the signal statistics and so reduces the *degree* of cyclostationarity.

5. DATA ACQUISITION IN THE PRESENCE OF TEMPORAL QUANTIZATION

We now return to the specific case of deterministic signal acquisition

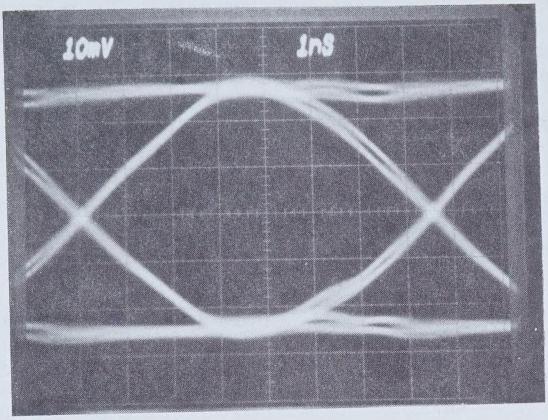


Figure 2 System eye diagram with i.s.i

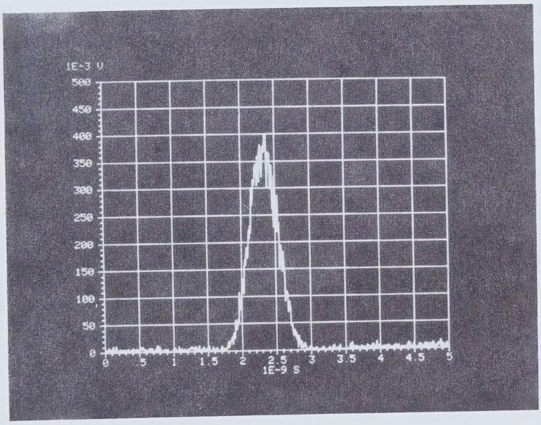


Figure 3 Illustrative measured jitter p.d.f

in the presence of additive noise and consider the limiting case of a large amount of signal averaging. This will reduce the noise in the acquired waveform. However, the time quantization error is equivalent to uniform random jitter over one effective sampling interval so an essentially deterministic impairment is introduced in the limit of substantial signal averaging. The aperture distortion for this important special case is also shown in table 1. Since averaging is inherent in the construction of a histogram estimator the foregoing argument applies also to measurements on a stochastic process: cyclic variation is subjected to a sinc(fΔ) filtering operation.

6. CONCLUSION

Some of the capabilities and limitations of equivalent-time acquisition/measurement systems have been considered and results have been presented relating to measurement precision which have implications for both the designers and users of such systems. We have noted that useful measurements relating to both deterministic signals and stochastic processes are possible and that a degree of non-stationarity - upto cyclostationarity - can be accommodated. This has been illustrated by way of certain specific measurement systems of importance for digital communications. These schemes provide for a more comprehensive statistical characterisation of digital communication signal processes than was formerly possible. In particular, detailed information relating to the important cyclostationary character can be obtained.

7. REFERENCES

1. Kareem A. and Thong T., "On the computation of the autocorrelation function of periodic signals by equivalent time sampling", Proc Conf Information Sciences and Systems; March 1982, Princeton, U.S.A. pp 349-355
2. O'Reilly J.J., "Equivalent-time signal acquisition and processing", Electron. Lett., 1981, 17, pp 911-912
3. Bennett W.R., "Statistics of digital transmission", B.S.T.J. 37 Nov. 1958, 1501-1542
4. O'Reilly J.J., "Temporal quantization noise", Electron. Lett., 1982, 18, pp 219-221
5. Cattermole K.W., "Principles of Pulse Code Modulation", (Iliffe, 1969)
6. Franks L.E., "Signal Theory", (Prentice Hall, 1969)
7. Cochrane P., Barley I. and O'Reilly J.J., "Direct jitter measurement technique", Electron. Lett., 1979, vol. 15, pp 774-776

Acknowledgement : This work was supported in part by SERC (U.K).

TONE JAMMER CANCELLATION IN A DIRECT SEQUENCE CORRELATOR

G. R. COOPER and E. H. COOPER
School of Electrical Engineering
Purdue University
W. Lafayette, IN 47907

In a direct sequence spread spectrum system employing short codes and a tapped-delay line correlator the processing gain is often insufficient to suppress the effects of strong tone jamming. Furthermore, the suppression that is actually achieved may be significantly less than that predicted by the processing gain because of code unbalance. In such cases it may be desirable to consider methods of achieving additional jammer cancellation in the correlator.

This paper presents the basic theory for cancelling an arbitrary number of tone jammers in an ideal tapped-delay line correlator by adjusting the tap weights. Computed results illustrate the theory and indicate the degree to which cancellation can be achieved for both single-tone jammers and multiple-tone jammers. In particular, almost perfect cancellation can be achieved for any offset frequency (i.e., the difference in frequency between the spread spectrum carrier and the jammer) and for jam-to-signal ratios as large as 50 dB when the tap weight values are adjusted to their optimum values for each offset frequency.

Since analog tapped-delay line correlators cannot be implemented readily, it is necessary to consider the effect of using tap weight values that have been quantized. In the present study this is accomplished by quantizing the optimum continuous values of tap weights for each offset frequency. Degradation in jammer cancellation is investigated as a function of the number of quantizing levels and the jam-to-signal ratio. Since analytical results are difficult to obtain for these cases, computer simulation is employed to obtain numerical values. This simulation indicates that 64 uniformly spaced quantizing levels are sufficient to achieve acceptable performance for jam-to-signal ratios as large as 50 dB. Since code polarity is known a priori, this number of quantizing levels requires only five bits of control information. It also appears that one less bit of control information is required for every 10 dB reduction in jam-to-signal ratio.

The optimum tap weights are critically dependent upon the jamming frequencies. Since these frequencies are not known a priori, it is necessary to estimate them from measurements of the received signal. Estimates of the jam-to-signal ratio and the receiver noise level are also required, but these are of lesser importance.

A MAP APPROACH TO JOINT PHASE TRACKING-SYMBOL ESTIMATION FOR MULTI-H SIGNALS USING THE VITERBI ALGORITHM

JOHN M. LIEBETREU AND RODGER E. ZIEMER
Electrical Engineering Department
University of Missouri-Rolla
Rolla, Missouri

ABSTRACT

A technique for obtaining the joint estimate of carrier phase and data is applied to multi-h signals. A MAP estimator structure is developed and used in a computer simulation to track a normal random walk phase jitter process in conjunction with symbol decoding via the Viterbi algorithm.

INTRODUCTION

The acquisition of carrier phase is a primary concern in digital data transmission systems. Together with symbol timing recovery, carrier phase tracking is a critical function in demodulating the received signal with a minimum performance loss. This is especially true for systems such as QPSK and MSK, where accurate estimation of the signal phase is essential to symbol decoding.

Another signalling format which depends on accurate knowledge of the signal phase for symbol decoding is the class of constant-envelope digital phase or frequency modulation techniques known as multi-h [1]. The signal space coordinates of a multi-h signal with carrier phase error are related by a rotational transformation to the signal space coordinates without carrier phase error. Since the phase process varies with time, the rotational transformation is dynamic in nature. Knowledge of this transformation is the essence of the data estimation problem, since it provides the proper signal space coordinate set for symbol decoding with the Viterbi algorithm. This paper formulates the joint estimation, by means of the Viterbi algorithm, of phase and data for multi-h modulated signals using a statistical estimation approach.

FORMULATION OF THE ESTIMATION PROCEDURE

In order to use the Viterbi algorithm for joint estimation of phase and data, the problem is formulated within a multiparameter statistical estimation framework [2], [3]. The fact that multi-h modulated signals are invariant to modulo-2π phase translations provides the basis for modeling the phase jitter as a modulo-2π phase process.

Phase jitter is often modeled as a Wiener-Levy process:

$$\frac{d\phi(t)}{dt} = w(t) \quad \text{for } t \geq 0 \tag{1}$$

where w(t) represents a white noise process. The discrete time analog of this process is

$$\phi_{k+1} = \phi_k + w_{k+1} \qquad k = 1, 2, . . . \tag{2}$$

where $\{w_k\}$ is a sequence of independent, identically distributed (i.i.d.) random variables with an even pdf. If $\{w_k\}$ is $N_{w_k}(0,\sigma_w^2)$ (Gaussian with zero mean and variance σ_w^2) then $\{\phi_k\}$ is the normal random walk. Denoting the modulo-2π version of this process as

$$\tilde{\phi}_{k+1} = \tilde{\phi}_k \oplus \tilde{w}_k \tag{3}$$

where $\oplus$ represents modulo-2π addition, a new process is defined as

$$\overline{\phi}_{k+1} = \tilde{\phi}_k + w_k \tag{4}$$

where $\tilde{\phi}_{k+1}$ is the modulo-2π version of $\overline{\phi}_{k+1}$. Then the conditional density of $\overline{\phi}_{k+1}$ given $\tilde{\phi}_k$ is $h(\overline{\phi}_{k+1} - \tilde{\phi}_k)$ where $h(\cdot)$ represents the pdf of w_k. Since $\tilde{\phi}_{k+1}$ is the modulo-2π version of $\overline{\phi}_{k+1}$

$$f(\phi_{k+1} | \phi_k) = \sum_{i=-\infty}^{\infty} h(\tilde{\phi}_{k+1} - \tilde{\phi}_k - 2\pi i) \overset{\Delta}{=} g(\tilde{\phi}_{k+1} - \tilde{\phi}_k). \tag{5}$$

Now suppose the data sequence is represented by $\{a_k\}_1^K$ and the phase sequence is $\{\phi_k\}_1^K$. Then the multi-h transmitter encodes $\{a_k\}_1^K$ into $\{\theta_k\}_1^K$ where $\{\theta_k\}$ obeys some special properties. The received sequence, using $s_k = e^{j\theta_k}$, is

$$r_k = e^{j\theta_k} \cdot e^{j\phi_k} + n_k \tag{6}$$

where $\{r_k\}_1^K$ and $\{n_k\}_1^K$ are complex. This is written as

$$r_k = e^{j\psi_k} + n_k \tag{7}$$

where the total phase is

$$\psi_k \hat{=} \theta_k + \phi_k \tag{8}$$

and

$$\theta_k = 2\pi \int_{-\infty}^{kT} \sum_{j=-\infty}^{\infty} a_j h_j \, g(\tau - jT) d\tau \tag{9}$$

is the data carrying phase that depends upon $g(\cdot)$, a frequency pulse. The important parameter for multi-h is actually $\Delta\theta_k$, the phase change occurring over the k^{th} symbol interval. The joint density to be maximized in the maximum *a posteriori* estimation problem is written as

$$f^K = \prod_{k=1}^{K} f(\psi_k, \Delta\theta_k | \{\psi_j\}_{j=1}^{k-1}, \{\Delta\theta_k\}_{j=1}^{k-1}) \, N_{r_k}(e^{j\psi_k}, \sigma_n^2) \tag{10}$$

where the maximum is found over all possible sequences $\{\psi_k\}_1^K$ and $\{\Delta\theta_k\}_1^K$, and

$$f(\psi_1,\Delta\theta_1 \,|\, \{\cdot\}, \{\cdot\}) = f(\psi_1,\Delta\theta_1) \tag{11}$$

Rewriting of the total phase process as

$$\begin{aligned}\psi_k &= \phi_k + \theta_k = \phi_{k-1} + w_k + \theta_{k-1} + \theta_k - \theta_{k-1} \\ &= \psi_{k-1} + \Delta\theta_k + w_k\end{aligned} \tag{12}$$

allows a simplification of f^K. Rewrite the conditional pdf $f(\psi_k,\, \Delta\theta_k \,|\, \{\cdot\}, \{\cdot\})$ and assume that timing of the modulation index sequence is available in the receiver, implying that $f(\Delta\theta_k \,|\, \{\cdot\}, \{\cdot\}) = \frac{1}{2}$. This yields

$$f(\psi_k,\, \Delta\theta_k \,|\, \{\cdot\}, \{\cdot\}) = \tfrac{1}{2}\, f(\psi_k \,|\, \{\cdot\}, \{\cdot\}) \tag{13}$$

Applying the argument used to obtain (5) gives the result that

$$f(\psi_k \,|\, \{\psi_j\}_1^{k-1},\ \{\Delta\theta_j\}_1^k) = g(\psi_k - \psi_{k-1} - \Delta\theta_k) \tag{14}$$

and the joint density to be maximized is

$$f^K = \prod_{k=1}^{K} N_{r_k}(e^{j\psi_k}, \sigma_n^2)\, (\tfrac{1}{2})\, g(\psi_k - \psi_{k-1} - \Delta\theta_k) \tag{15}$$

If the phase increment w_k is assumed to be Gaussian, $g(\cdot)$ is unimodal and even, so f^K is maximized by choosing

$$\hat{\Delta\theta}_k = <[\hat{\psi}_k - \hat{\psi}_{k-1}]> \tag{16}$$

where $<[x]>$ denotes the closest value of $2\pi \int_{(k-1)T}^{kT} a_j h_j g(\tau - kT)\,d\tau$ to $(\hat{\psi}_k - \hat{\psi}_{k-1})$.

From this point the dynamic programming equations are found in a straightforward manner [4]:

$$\begin{aligned}
&\text{(a)}\quad \max_{\{\psi_k\}_1^K} \Gamma_K \\
&\text{(b)}\quad \Gamma_{1j} = \frac{-1}{2\sigma_n^2}\left|r_{1j} - e^{j\psi_{1j}}\right|^2 + \ln g(\psi_{1j} - <[\psi_{1j}]>) \\
&\text{(c)}\quad P_{kj} = \frac{-1}{2\sigma_n^2}\left|r_{kj} - e^{j\psi_j}\right|^2 + \ln g\{(\psi_{kj} - \psi_{k-1,j}) - \\
&\qquad\qquad <[\psi_{kj} - \psi_{k-1,j}]>\} \\
&\text{(d)}\quad \Gamma_{kj} = \Gamma_{k-1,j} + P_{kj}
\end{aligned} \tag{17}$$

For the multi-h receiver shown in Figure 1 the notation $|\cdot|^2$ in (b) and (c) above represents the norm squared $||\cdot||^2$ in the discrete-time inner product space. The correlation receiver actually makes use of the inner product

$$\bar{\xi}_{kj} \triangleq \int_{(k-1)T}^{kT} [e(t) - \frac{1}{\sqrt{T}} e^{j\psi(t)}][e(t) - \frac{1}{\sqrt{T}} e^{j\psi(t)}]^{*} dt \qquad (18)$$

where e(t) represents the complex baseband envelope of r(t). After the appropriate substitutions and trigonometric expansions this expression is found to be equivalent to (see Appendix)

$$\bar{\xi}^{(1)}_{kj} = 2\sqrt{2}\;[A_2' \sin\theta_{kj} - A_1' \cos\theta_{kj}] \qquad (19)$$

for a data '1' and

$$\begin{aligned}\bar{\xi}^{(-1)}_{kj} &= 2\sqrt{2}\;[A_1' \, CQH_k + A_2' \, SQH_k + A_3' \, DN_k)\cos\theta_{kj} \\ &\quad + (A_1' \, SQH_k - A_2' \, CQH_k - A_4' \, DN_k)\sin\theta_{kj}] \end{aligned} \qquad (20)$$

for a data '0'. In these expressions the vector $\underline{A}'$ represents the signal space coordinates after the rotational transformation has been applied by the random phase and θ_{kj} represents the data phase at the beginning of the kth symbol interval on path j. The orthonormalization constants, also shown in Figure 1, are computed by the Gram Schmidt procedure to obtain a set of orthonormal basis functions for the signal space, given the correlation functions shown [5].

RESULTS

A block diagram of the computer simulation program used to obtain performance data is shown in Figure 2. Figure 3 shows the probability of error as a function of signal to noise ratio for the case h = ½ with the phase space quantized to 32 levels. Notice that the expected improvement in performance as phase jitter variance is decreased is verified by the plots for differing values of the variance σ_w^2. Figure 4 shows probability of error as a function of signal-to-noise ratio for the two-h case where $h_1 = \frac{3}{8}$ and $h_2 = \frac{4}{8}$. The phase space is also quantized to 32 levels for these results. In Figure 5 three plots are shown of probability of error as a function of the phase jitter variance σ_w^2. These plots also show the effect of phase space quantization. Notice that the performance improvement gained by a more finely quantized phase space is not significant. When the square root of the phase jitter variance is smaller than the quantization intervals of the phase space a noticeable degradation in performance occurs. This is due to measurement error introduced in estimating a quantity which is smaller than the device used to express the value of this estimate. This expected result confirms the need to choose a phase space quantization level that is compatible with the phase jitter variance.

CONCLUSION

A method is described for joint phase tracking and symbol decoding of a multi-h signal using a correlation receiver and the Viterbi algorithm. A computer simulation is presented for use in obtaining performance data and comparisons to other "Viterbi-like" algorithms for phase tracking and symbol decoding.

ACKNOWLEDGEMENT

This work was performed with partial support from Motorola, Inc., Government Electronics Group.

APPENDIX

The orthonormalization constants given in equations (19) and (20) and shown in Figure 1 are defined as follows:

$$CQH_k \triangleq \frac{1}{T}\int_0^T \cos\,[4\pi h_k q(t)]dt$$

$$SQH_k \triangleq \frac{1}{T}\int_0^T \sin\,[4\pi h_k q(t)]dt$$

$$DN_k = [1 - (CQH_k)^2 - (SQH_k)^2]^{\frac{1}{2}}$$

where $q(t) \triangleq \int_{-\infty}^{t} g(\tau)d\tau$; $0<t<T$ and $g(t)$ is the frequency pulse. Also,

$$XIC \triangleq \sqrt{\frac{2}{T}}\int_0^T I(t)\,\cos\,[2\pi h_k q(t)]dt$$

$$XIS \triangleq \sqrt{\frac{2}{T}}\int_0^T I(t)\,\sin\,[2\pi h_k q(t)]dt$$

$$XQC \triangleq \sqrt{\frac{2}{T}}\int_0^T Q(t)\,\cos\,[2\pi h_k q(t)]dt$$

$$XQS \triangleq \sqrt{\frac{2}{T}}\int_0^T Q(t)\,\sin\,[2\pi h_k q(t)]dt$$

where $e(t) = I(t) + jQ(t)$.

REFERENCES

1. T. Aulin and C. E. Sundberg, "Continuous Phase Modulation (CPM), Part I: Full Response Signalling", IEEE Trans. Commun., Vol. COM-29, pp. 196-209, March 1981.

2. G. Ungerboeck, "New Application for the Viterbi Algorithm: Carrier Phase Tracking in Synchronous Data Transmission Systems", in Proc. Nat. Telecomm. Conf., 1974, pp. 734-738.

3. O. Macchi and L. L. Scharf, "A Dynamic Programming Algorithm for Phase Estimation and Data Decoding on Random Phase Channels," IEEE Trans. Inform Theory, Vol. IT-27, pp. 581-595, Sept. 1981.

4. G. D. Forney, "The Viterbi Algorithm", Proc. IEEE, Vol. 61, pp. 268-278, March 1973.

5. V. K. Bhargava, D. Haccoun, R. Matyas, and P. Nuspl, Digital Communications by Satellite, pp. 181-188, New York: Wiley, 1981.

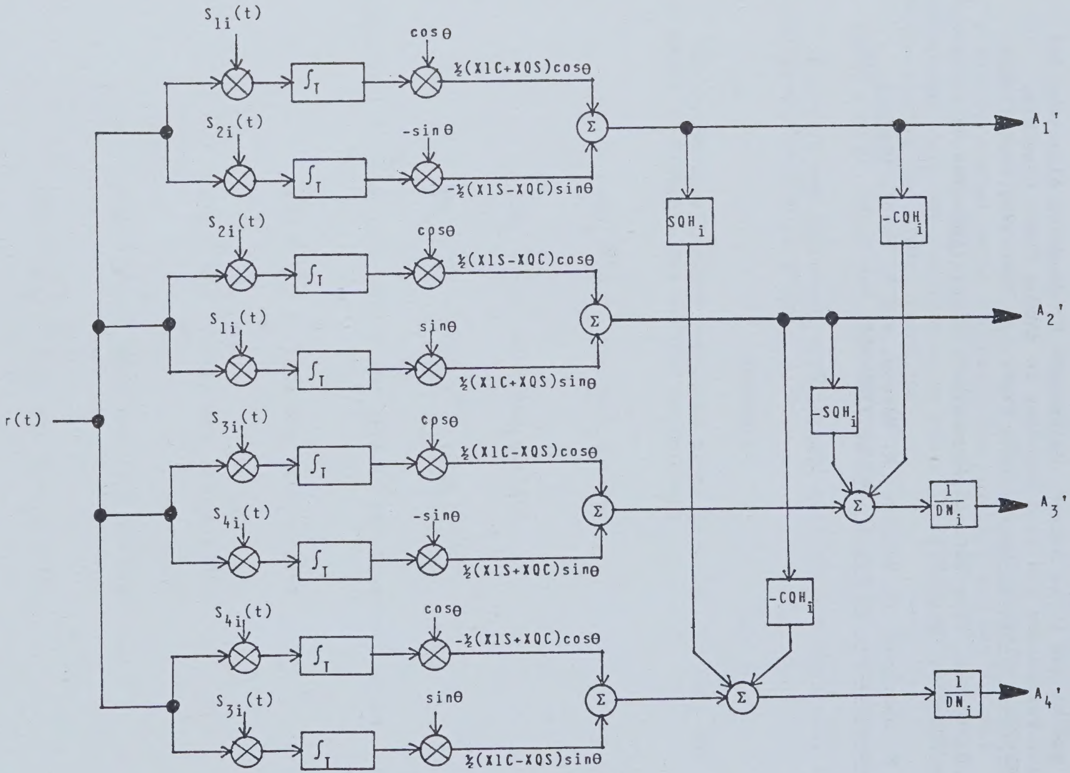

$S_{1i}(t)$
$S_{2i}(t)$
$S_{2i}(t)$
$S_{1i}(t)$
$S_{3i}(t)$
$S_{4i}(t)$
$S_{4i}(t)$
$S_{3i}(t)$
r(t)
$\int_T$
cosθ
-sinθ
cosθ
sinθ
cosθ
-sinθ
cosθ
sinθ
½(X1C+XQS)cosθ
-½(X1S-XQC)sinθ
½(X1S-XQC)cosθ
½(X1C+XQS)sinθ
½(X1C-XQS)cosθ
½(X1S+XQC)sinθ
-½(X1S+XQC)cosθ
½(X1C-XQS)sinθ
Σ
SQH_i
$-CQH_i$
$-SQH_i$
$-CQH_i$
$\frac{1}{DN_i}$
$\frac{1}{DN_i}$
A_1'
A_2'
A_3'
A_4'

Figure 1. Correlation Receiver for Multi-h Signals Showing the Effect of Carrier Phase Error.

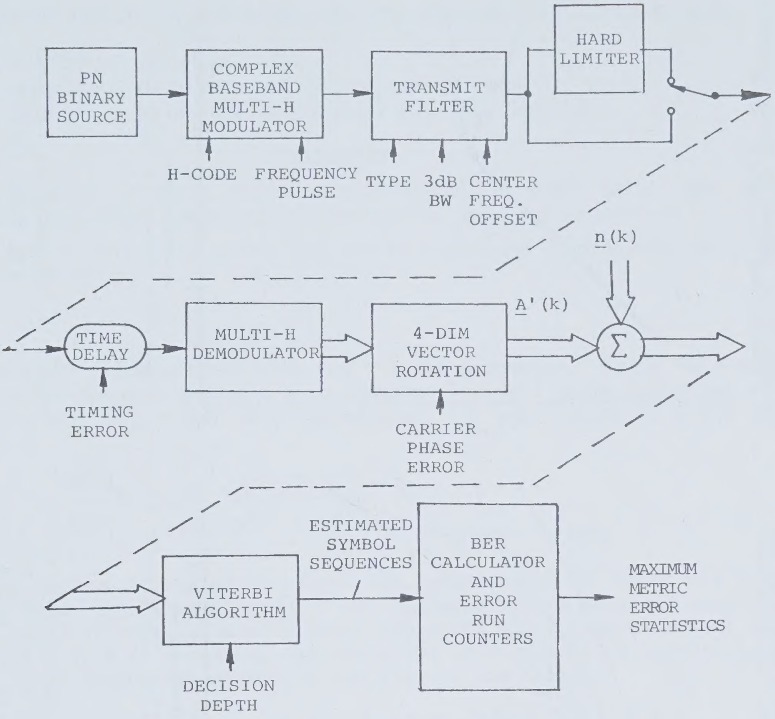

PN BINARY SOURCE
COMPLEX BASEBAND MULTI-H MODULATOR
TRANSMIT FILTER
HARD LIMITER
H-CODE
FREQUENCY PULSE
TYPE
3dB BW
CENTER FREQ. OFFSET
n(k)
A'(k)
TIME DELAY
MULTI-H DEMODULATOR
4-DIM VECTOR ROTATION
Σ
TIMING ERROR
CARRIER PHASE ERROR
ESTIMATED SYMBOL SEQUENCES
VITERBI ALGORITHM
BER CALCULATOR AND ERROR RUN COUNTERS
MAXIMUM METRIC ERROR STATISTICS
DECISION DEPTH

Figure 2. Block Diagram of the Computer Simulation Program.

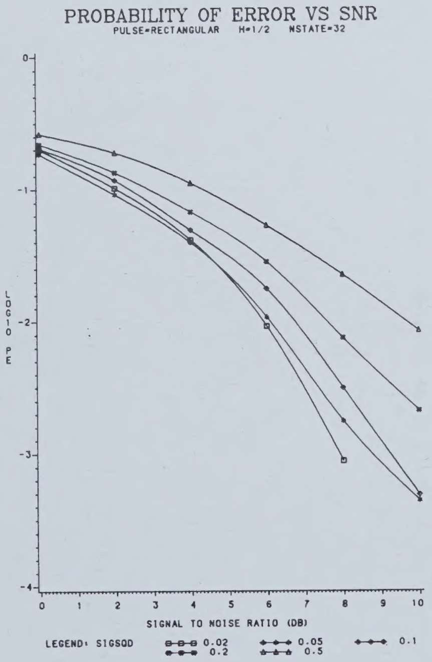


Figure 3. Probability of Error vs. Signal to Noise Ratio for h=½.

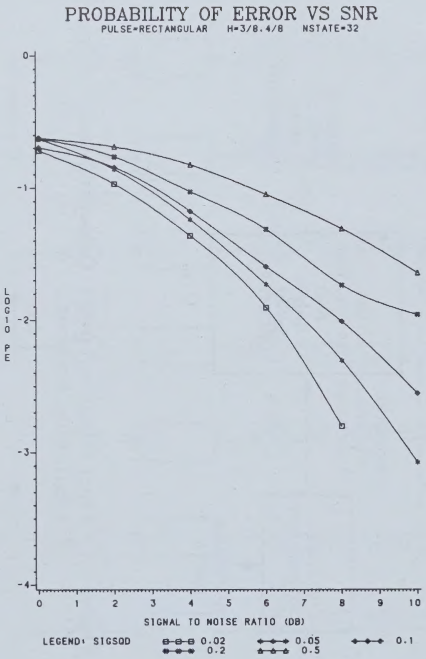


Figure 4. Probability of Error vs. Signal to Noise Ratio for h=3/8, 4/8.

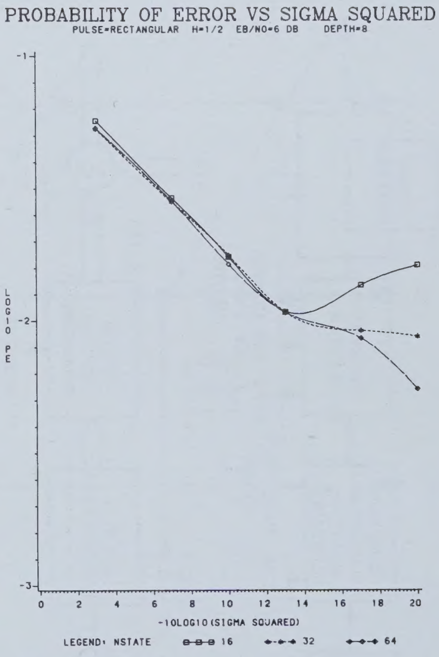


Figure 5. Probability of Error vs. Phase Jitter Variance σ_w^2.

THE NOISE PENALTY IN APD RECEIVERS OPERATING WITH NON-OPTIMUM GAIN

LAWRENCE J MAYES
Department of Electrical Engineering Science
University of Essex, Colchester, CO4 3SQ, England

ABSTRACT

Contours showing impairment in signal to noise ratio for avalanche optical receivers operating at non-optimum gain are presented.

The excess noise factor (F) for an avalanche photodiode (APD) is often approximated by:

$$F = m^x \tag{1}$$

where $x \approx 0{\cdot}3$ for silicon APDs and ≈ 1 for germanium. Using this expression the noise penalty suffered by a receiver using a diode operating at a non-optimum gain (m) rather than the optimum (m_o) can readily be found [1]:

$$\frac{S/N|_m}{S/N|_{m_o}} = \frac{r^2(1 + x/2)}{r^{2+x} + x/2} \qquad \text{where } r = m/m_o \tag{2}$$

A closer approximation for F has been derived [2] which has been shown to be good experimentally [3]. In the model it is assumed that the ratio of ionisation coefficients of holes and electrons in the avalanche region of the diode is k (a constant, typically $k = 0{\cdot}02$ in silicon and $0{\cdot}5$ in germanium). The resulting expression for F is of the form:

$$F = km + (1 - k)(2 - 1/m) \tag{3}$$

Therefore an improved indication of impairment may be obtained by repeating the procedure using equation (3). The impairment for non-optimum gain is now given by:

$$\frac{S/N|_m}{S/N|_{m_o}} = r^2 \left[\frac{2m_o(km_o+(1-k)(2-1/m_o))+(k(m_o^{\,2}-1)+1)}{2m_o r^2(rkm_o+(1-k)(2-1/(rm_o)))+(k(m_o^{\,2}-1)+1)}\right] \tag{4}$$

We note that both sensitivity expressions produce similar predictions for values of k and x close to unity; indeed, the expressions become equivalent when $k = x = 1$.

By way of example, contour plots showing lines of equal impairment (spaced at 1 dB intervals) are reproduced in figures 1 and 2; these apply to silicon APD receivers. The plots for germanium (for which values of $k = 0{\cdot}5$ and $x = 1$ may be assumed) are almost identical (to within 0·2 dB) for moderate gain errors and consequently not reproduced here.

In summary, we can conclude that equation (2) predicts a lower sensitivity of signal to noise ratio to avalanche gain variation for practical silicon APD receivers than the more realistic prediction provided by equation (4) which is based on a more precise expression for the diode noise factor. In the case of APDs fabricated from germanium or materials with similar k values, differences between the two predictions may be small enough to be neglected and under these circumstances equation (2) is applicable.

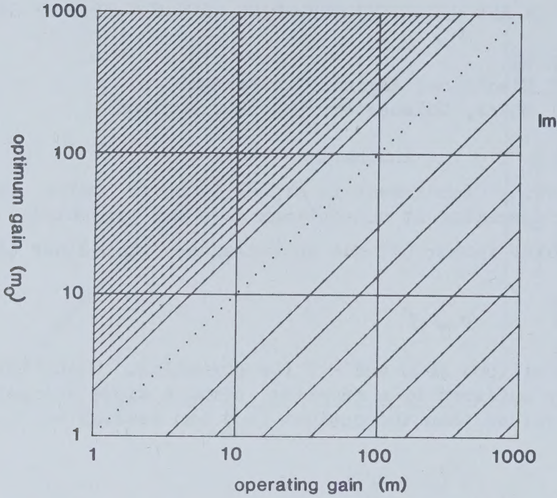


Figure 1:
Impairment Contours
x = 0.3

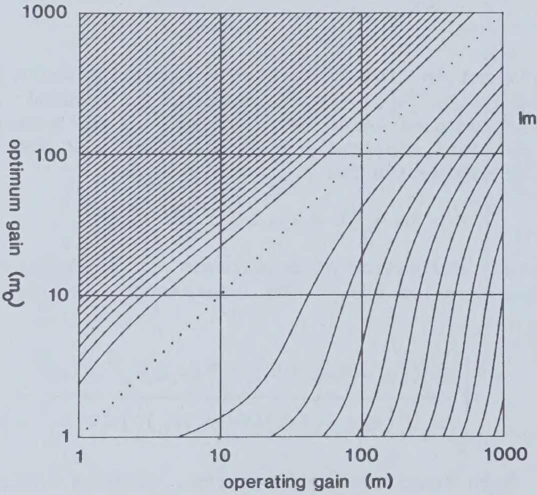


Figure 2:
Impairment Contours
k = 0.02

REFERENCES

1 M M IBRAHIM and M A ALHAIDER: 'Sensitivity of avalanche-photodiode optical receivers for avalanche gain'. AEU (Japan), February 1982, pp 153-155.

2 R J McINTYRE: 'Multiplication noise in uniform avalanche diodes'. IEEE Transactions on Electron Devices, vol ED-13 no 1, January 1966, pp 164-168.

3 J CONRADI: 'The distribution of gains in uniformly multiplying avalanche photodiodes: experimental'. IEEE Transactions on Electron Devices, vol ED-19 no 6, June 1972, pp 713-718.

Acknowledgement: Financial support for the above work was provided by the SERC (UK).

STABILITY ANALYSIS OF FIXED-POINT DIGITAL FILTERS USING COMPUTER GENERATED LYAPUNOV FUNCTIONS

Kelvin T. Erickson
Fisher Controls Co.
Marshalltown, Iowa 50158

Anthony N. Michel
Iowa State University
Ames, Iowa 50011

ABSTRACT

We demonstrate the applicability of the constructive stability algorithm of Brayton and Tong in the stability analysis of fixed-point digital filters. In the present paper, we consider two direct form filter structures. We compare our results with existing ones which deal with the stability of fixed-point digital filters. Several of the present results are new while some of them constitute improvements over existing results. It is emphasized that whereas the existing results are obtained by several diverse methods, the present results are determined by one unified approach.

INTRODUCTION

In two recent papers, Brayton and Tong [1], [2] established some significant results which make it possible to construct computer generated Lyapunov functions to analyze the stability of nonlinear systems. These results were subsequently extended in several ways, making it possible to estimate the domain of attraction of an equilibrium [3] and to apply the constructive algorithm to high-dimensional systems [4].

In this paper, we apply the constructive algorithm of Brayton and Tong to the stability analysis of second order direct form fixed-point digital filters. Coupled form, wave and lattice filters are considered in [5]. Nonlinearities which we encounter in our analysis of these filters include several types of fixed-point quantization effects and overflow effects. The results which we obtained yield conditions (in the parameter plane) under which the digital filters which we consider are globally asymptotically stable (g.a.s.), and as such, do not possess zero-input limit cycles. We compare these results with several corresponding existing results ([6]-[8]) which are concerned either with the existence or nonexistence of limit cycles or with the global asymptotic stability of discrete systems.

CONSTRUCTIVE STABILITY ALGORITHM

In two papers [1] and [2], Brayton and Tong present an algorithm to construct Lyapunov functions to establish the stability and global asymptotic stability of the equilibrium $x=0$ of dynamical systems described by ordinary differential equations and also by difference equations. The basic philosophy of their method is to determine the stability of an appropriate set of matrices associated with the system in question and then deduce the stability of the equilibrium of the given system. To utilize this algorithm, consider systems of the form

$$x(k+1) = g[x(k)]. \tag{1}$$

We rewrite the system of equations (1) as

$$x(k+1) = M(x(k))x(k) \tag{2}$$

where $M(x(k))$ is chosen so that $M(x(k))x(k) = g[x(k)]$. For every $x \in R^n$, $M(x)$ will be a real $n \times n$ matrix. If we let $\underline{M}$ denote the set of all matrices, M, obtained by varying x in $M(x)$ over all allowable values, then we can rewrite (2) equivalently as

$$x(k+1) = M_k x(k), \ M_k \varepsilon \underline{M}. \tag{3}$$

Brayton and Tong [1], [2] show that the equilibrium x=0 of (3) is stable (globally asymptotically stable) if the set of matrices $\underline{M}$ is stable (asymptotically stable). In the following, we give a brief summary of the results of Brayton and Tong.

We call a set $\underline{A}$ of n×n real matrices stable if for every neighborhood of the origin $U \subset R^n$, there exists another neighborhood of the origin $V \subset R^n$ such that for every $M \varepsilon \underline{A}'$, we have $MV \subseteq U$. Here, $\underline{A}'$ denotes the multiplicative semigroup generated by a set $\underline{A}$ and $MV = \{u \varepsilon R^n: u=Mv, \ v \varepsilon V\}$.

In [1] it is shown that if a set $\underline{A}$ of n×n matrices is stable, then there exists a vector norm $|\cdot|_w$ such that $|Mx|_w \leqslant |x|_w$ for all $M \varepsilon \underline{A}$ and $x \varepsilon R^n$. The vector norm $|\cdot|_w$ defines a Lyapunov function for $\underline{A}$, i.e., it defines a function v with the property

$$v(Mx) \leqslant v(x), \quad \text{for all} \quad M\varepsilon\underline{A} \quad \text{and} \quad x\varepsilon R^n.$$

Next, we call a set of matrices $\underline{A}$ asymptotically stable if there exists a number $\rho > 1$ such that $\rho\underline{A}$ is stable. (The set $\rho\underline{A}$ is obtained by multiplying every member of $\underline{A}$ by ρ.) Thus, if $\rho\underline{A}$ is stable, then there exists a vector norm $|\cdot|_w$ such that

$$|Mx|_w < |\rho Mx|_w \leqslant |x|_w \quad \text{for all} \quad M\varepsilon\underline{A} \quad \text{and} \quad x\varepsilon R^n.$$

The vector norm $|\cdot|_w$ defines a Lyapunov function with the property

$$v(Mx) < v(x), \quad \text{for all} \quad M\varepsilon\underline{A} \quad \text{and} \quad x\varepsilon R^n.$$

Thus, given the system (3), if we can show that $\underline{M}$ is asymptotically stable, then we can conclude that the system (3) is globally asymptotically stable, by Lyapunov stability theory (see, e.g., [12]).

In [1] and [2], a constructive algorithm is given to determine whether a finite set of real matrices is either stable, by constructing the Lyapunov function, or unstable by determining that the vector norm Lyapunov function does not exist.

The set of matrices $\underline{M}$ given in (3) consists in general of an infinite number of matrices. However, in [1] it is shown that we need only work with the extreme matrices of the set, $E(\underline{A})$, to get an indication of the stability of $\underline{A}$. Thus, we can use a finite set, $E(\underline{A})$, to analyze the stability of the infinite set, $\underline{A}$.

APPLICATION OF THE CONSTRUCTIVE ALGORITHM TO THE STABILITY ANALYSIS OF DIGITAL FILTERS

Nonlinearities in Digital Filters

In fixed-point digital filters, the representation of signals must necessarily have finite precision. Multiplications and additions performed in the digital filter generally lead to an increase in the wordlength required for the result of the operation. In a recursive digital filter, a wordlength reduction of the result is necessary to prevent the wordlength of the signals from increasing indefinitely. A quantization nonlinearity is produced when the result of a multiplication is reduced in wordlength. Quantization only affects the least significant bits. An overflow nonlinearity occurs when the sum of two numbers is modified to fall back within the representable range. In general, the overflow nonlinearity changes the most significant bits and the least significant bits of a fixed-point number. These two types of nonlinearities are well described in the literature (see e.g., [11]) and therefore, are not discussed here. The characteristics of the quantization and overflow nonlinearities that we consider are shown in Figure 1. Other types of overflow nonlinearities are considered in [5].

These nonlinearities may be viewed as belonging to a sector [k, K]. Thus, if f(•) denotes a given nonlinearity, then

$$k\sigma^2 \leqslant \sigma f(\sigma) \leqslant K\sigma^2 \quad \text{for all} \quad \sigma \varepsilon R,$$

where k and K are constants such that $-\infty < k \leqslant K < \infty$. Under this assumption, we view the quantization nonlinearities as belonging to the sector $[0, k_q]$ where

$$k_q = \begin{cases} 1 & \text{truncation} \\ 2 & \text{roundoff.} \end{cases} \tag{4}$$

Overflow is represented as belonging to the sector $[k_o, 1]$ where

$$k_o = \begin{cases} 0 & \text{saturation} \\ -1 & \text{two's complement.} \end{cases} \tag{5}$$

When the quantization and overflow functions are executed simultaneously, then the combined nonlinear operation is represented as belonging to the sector $[k_o, k_q]$.

General Digital Filter

Since we consider only second order digital filters, the matrix M in (2) may be written as

$$M(x(k)) = \begin{bmatrix} a(x(k)) & b(x(k)) \\ c(x(k)) & d(x(k)) \end{bmatrix}.$$

We assume that the elements of M satisfy the inequalities

$$a_1 < a(x(k)) < a_2 \qquad c_1 < c(x(k)) < c_2$$
$$b_1 < b(x(k)) < b_2 \qquad d_1 < d(x(k)) < d_2$$

where a_i, b_i, c_i and d_i, i=1,2 are constants. Then, the extreme matrices of set $\underline{M}$ in (3) are obtained as

$$E(\underline{M}) = \left\{ \begin{bmatrix} a_i & b_j \\ c_k & d_\ell \end{bmatrix}, \ i,j,k,\ell = 1,2 \right\}. \tag{6}$$

We need only determine that $E(\underline{M})$ is asymptotically stable to determine that the digital filter described by (1) is globally asymptotically stable. If the set $\underline{M}$ is unstable, then we can draw no conclusion about the stability of the filter. Using the constructive algorithm, we show that $E(\underline{M})$ is asymptotically stable by choosing $\rho = 1.0000001$ and then showing that $\rho\underline{M}$ is stable by using the constructive algorithm. Since the constructive algorithm is used to show that the equilibrium x=0 of a given digital filter (1) is globally asymptotically stable, then in particular, no limit cycles will exist in this digital filter.

To implement the constructive algorithm, we evaluate the set of extreme matrices (6) at each point in a sufficiently fine grid in the parameter plane. We used a grid of 0.02 units for the results presented here. The computer programs that implement the constructive algorithm are described in detail in [5].

Direct Form Digital Filter

In this section, we show how to apply the Brayton-Tong constructive algorithm to the stability analysis of the direct form digital filter. We present the particular nonlinear structures for this type of filter and then present the set of extreme matrices used by the constructive algorithm.

Since we only consider filters with zero input, the recursive parts of the direct form 1 structure and the direct form 2 structure are equivalent. When the linear second order direct form digital filter is implemented in fixed-point arithmetic, there are two possible ways of placing the quantization nonlinearity. Quantization can be performed immediately after each multiplication. This structure is shown in Figure 2, with Q representing a quantizer. Alternatively, the results of the two multiplications may be added with full precision and only one quantizer is needed. This structure is shown in Figure 3. For both possible quantizer configurations, the overflow nonlinearity, P, is placed after the adder as shown.

To develop the set of extreme matrices for the direct form filter with one quantizer, we consider the quantization and overflow nonlinearities together. With this assumption, the state equations are

$$\begin{aligned} x_1(k+1) &= f[ax_1(k) + bx_2(k)] \\ x_2(k+1) &= x_1(k) \end{aligned} \tag{7}$$

where f(•) is the combined quantization and overflow nonlinearity. When the state equations (7) are written in the form of (2), the matrix M(x(k)) is given by

$$M(x(k)) = \begin{bmatrix} \Phi(x)a & \Phi(x)b \\ 1 & 0 \end{bmatrix}$$

where

$$\Phi(x) = \frac{f(ax_1+bx_2)}{ax_1 + bx_2} .$$

Since the nonlinearity f(•) belongs to a sector, the function $\Phi(x)$ is bounded by constants, α_1 and α_2, such that $\alpha_1 \leqslant \Phi(x) \leqslant \alpha_2$. For the particular nonlinearities which we consider, we have $\alpha_1 = k_o$ and $\alpha_2 = k_q$ where k_q and k_o are defined by (4) and (5), respectively. The extreme matrices of the set $\underline{M}$ are

$$E(\underline{M}) = \left\{ \begin{bmatrix} \alpha_i a & \alpha_i b \\ 1 & 0 \end{bmatrix}, \quad i=1,2 \right\}. \tag{8}$$

Thus, for each point (a,b) in the parameter plane, there are two extreme matrices. If the overflow nonlinearity is absent, the set of extreme matrices is the same as the filter with a saturation overflow nonlinearity.

Following the same technique, the set of extreme matrices for the direct form filter with two quantizers is shown in [5] to be

$$E(M) = \left\{ \begin{bmatrix} \alpha_i a & \beta_j b \\ 1 & 0 \end{bmatrix}, \quad i,j = 1,2 \right\} \tag{9}$$

where $\alpha_1 = \beta_1 = k_q k_o$ and $\alpha_2 = \beta_2 = k_q$. Thus, for each point (a,b) in the parameter plane, there are four extreme matrices. If the overflow nonlinearity is absent, the set of extreme matrices is the same as the filter with a saturation overflow nonlinearity.

COMPARISON OF STABILITY RESULTS BY THE CONSTRUCTIVE ALGORITHM WITH EXISTING RESULTS

Direct Form, One Quantizer

To apply the constructive algorithm to the direct form filter with one quantizer (Figure 3), we use the extreme matrices determined by (8). The regions in the parameter plane where this digital filter is globally asymptotically stable for all of the combinations of nonlinearities that we consider in this paper are shown in Figure 4. The region where a particular filter is globally asymptotically stable is indicated as the unhatched region within the larger triangle. Since the linear filter is unstable for all parameters outside of this triangle, we are not interested in the nonlinear filter whose parameters fall outside of this region. The horizontally hatched region indicates the region where the set of extreme matrices is unstable because one of the eigenvalues of an extreme matrix has a magnitude greater than one. Vertical hatching indicates the remainder of the region where the set of extreme matrices is unstable, and thus, where we can draw no conclusion about the stability of the filter. These regions are symmetric about the b axis. The stability results obtained by the constructive algorithm for the overflow nonlinearity without quantization are the same as the results for the overflow and truncation nonlinearities combined.

For this filter with truncation quantization and no overflow, the constructive algorithm yields the same region in the parameter plane where the filter is globally asymptotically stable as the result of Claasen et al. [6] that shows the absence of limit cycles in this region. For the filter with a roundoff quantizer and no overflow, the constructive algorithm obtains a slightly larger region than Claasen et al. [7], which is indicated by the dashed line of Figure 4(c). For the filter without overflow, limit cycles have been found by others in all of the horizontally hatched region of Figure 4(c) [6]. If we consider the saturation overflow nonlinearity only, then the result by the constructive algorithm is more conservative than that of Ebert, Mazo and Taylor [8], who show that no limit cycles exist in that particular filter. However, the result obtained by the constructive algorithm applies to a class of nonlinearities instead of a single nonlinearity. The constructive algorithm yields the same region as [8] where limit cycles are absent for this filter with a single two's complement overflow. All of the stability results for this filter when quantization and overflow are considered together appear to be new.

Direct Form, Two Quantizers

To apply the constructive algorithm to the direct form filter with two quantizers (Figure 2), we use the extreme matrices determined by (9). The regions in the parameter plane where this digital filter is globally asymptotically stable for all of the combinations of nonlinearities that we consider in this paper are shown in Figure 5. Hatching has the same indication as in Figure 4. These regions are also symmetric about the b axis.

For the filter with truncation or roundoff quantization, and no overflow, the constructive algorithm obtains a less conservative result than the application of the absolute stability criterion of Jury and Lee [9]. The regions where this filter is globally asymptotically stable by [9] are within the dashed lines of Figures 5(a) and 5(c). For the filter without overflow, limit cycles have been found by others in all of the horizontally hatched region of Figure 5(a) and nearly all of the horizontally hatched region of Figure 5(c) [6]. The remainder of the results appear to be new.

CONCLUSION

The direct form fixed-point digital filter structures which we have analyzed demonstrate that the Brayton-Tong constructive algorithm is a powerful tool in the stability analysis of fixed-point digital filters. We have obtained new results for the structures which we have analyzed. We also improved upon existing results. Our results are more conservative only in one case. In that case, the existing result considers a specific nonlinearity, whereas the constructive algorithm obtains a stability result which applies to a class of nonlinearities.

Whereas existing methods of stability analysis are generally different for each particular filter structure, the constructive algorithm allows us to use one method to study the stability of nonlinear digital filter structures. We feel that this method should become a tool to be used in the evaluation of any proposed new filter structure.

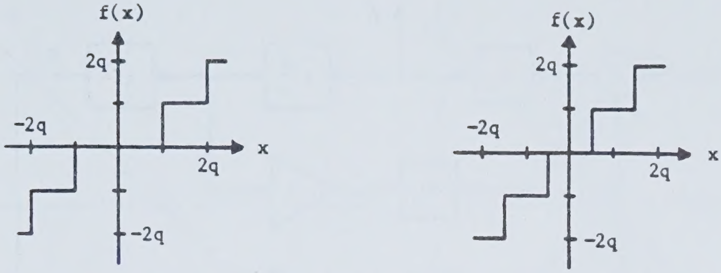
In Erickson [5], second order coupled form, wave and lattice filters are considered. There are many other digital filter structures one might want to analyze by the constructive algorithm. The constructive algorithm could also be applied to higher order filters either directly or by considering the higher order filter as an interconnection of lower order structures (see, e.g., [4],[10]).

REFERENCES

1. Brayton, R. K. and Tong, C. H., "Stability of Dynamical Systems: A Constructive Approach". IEEE Transactions on Circuits and Systems, CAS-26 (1979): 224-234.
2. Brayton, R. K. and Tong, C. H., "Constructive Stability and Asymptotic Stability of Dynamical Systems". IEEE Transactions on Circuits and Systems, CAS-27 (1980): 1121-1130.
3. Michel, A. N., Sarabudla, N. R. and Miller, R. K., "Stability Analysis of Complex Dynamical Systems: Some Computational Methods". Circuits, Systems and Signal Processing, 1 (1982): 171-202.
4. Michel, A. N., Miller, R. K. and Nam, B. H., "Stability Analysis of Interconnected Systems Using Computer Generated Lyapunov Functions". IEEE Transactions on Circuits and Systems, CAS-29 (1982): 431-440.
5. Erickson, K. T., "Stability Analysis of Fixed-Point Digital Filters Using a Constructive Algorithm". Ph.D. Dissertation, Iowa State University, Ames, Iowa, 1983.
6. Claasen, T. A. C., Mecklenbräuker, W. F. G. and Peek, J. B. H., "Effects of Quantization and Overflow in Recursive Digital Filters". IEEE Transactions on Acoustics, Speech and Signal Processing, ASSP-24 (1976): 517-529.
7. Claasen, T., Mecklenbräuker, W. F. G. and Peek, J. B. H., "Frequency Domain Criteria for the Absence of Zero-Input Limit Cycles in Nonlinear Discrete-Time Systems, with Applications to Digital Filters". IEEE Transactions on Circuits and Systems, CAS-22 (1975): 232-239.
8. Ebert, P. M., Mazo, J. E. and Taylor, M. G., "Overflow Oscillations in Digital Filters". Bell System Technical Journal, 48 (1969): 2999-3020.
9. Jury, E. I. and Lee, B. W., "The Absolute Stability of Systems with Many Nonlinearities". Automation and Remote Control, 26 (1965): 943-961.
10. Michel, A. N. and Miller, R. K., Qualitative Analysis of Large Scale Dynamical Systems. New York: Academic Press, 1977.
11. Oppenheim, A. V. and Schafer, R. W., Digital Signal Processing Englewood Cliffs, N. J.: Prentice-Hall, 1975.
12. Miller, R. K. and Michel, A. N., Ordinary Differential Equations. New York: Academic Press, 1982.

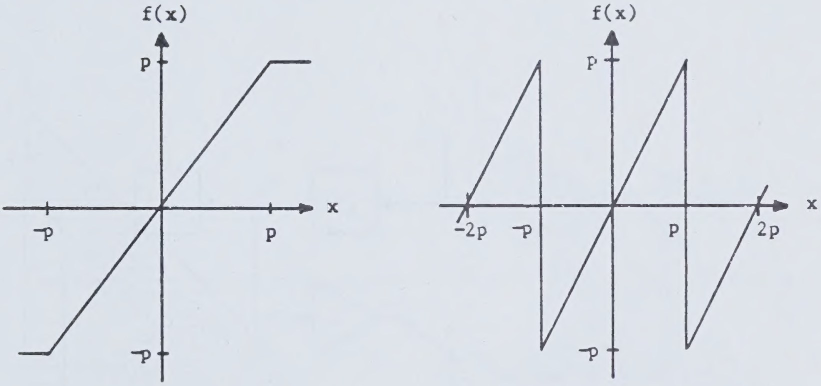
ACKNOWLEDGEMENT

This work was supported in part by the National Science Foundation under Grant ECS-81-00690 and the Engineering Research Institute, Iowa State University, Ames, Iowa.



(a) Magnitude truncation quantization

(b) Roundoff quantization



(c) Saturation overflow

(d) Two's complement overflow

Figure 1. Fixed-point digital filter nonlinearity characteristics

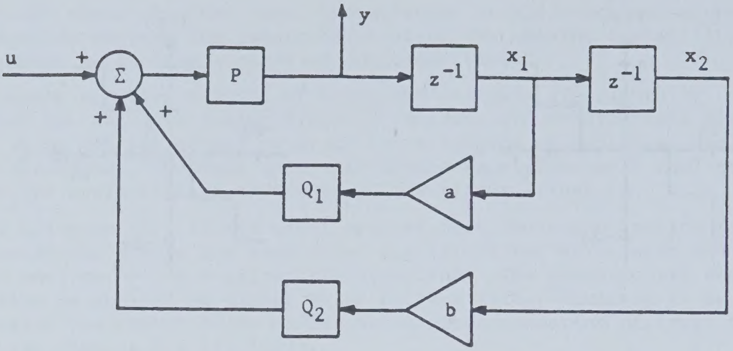

y
u
+
Σ
P
+
+
z^{-1}
x1
z^{-1}
x2
Q1
a
Q2
b

Figure 2. Direct form digital filter with two quantizers

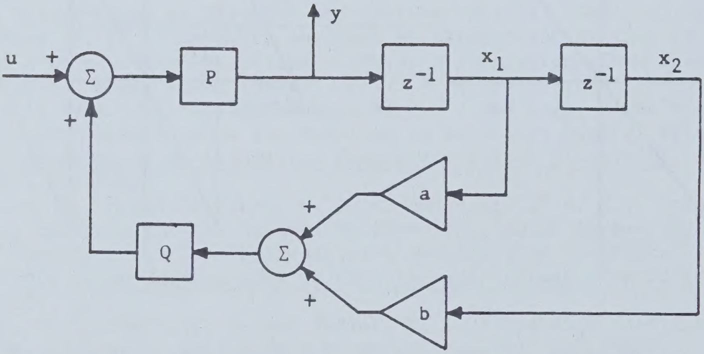
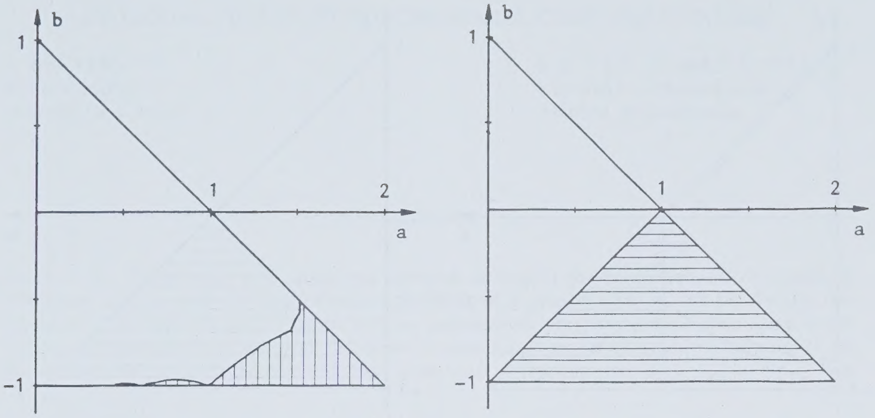
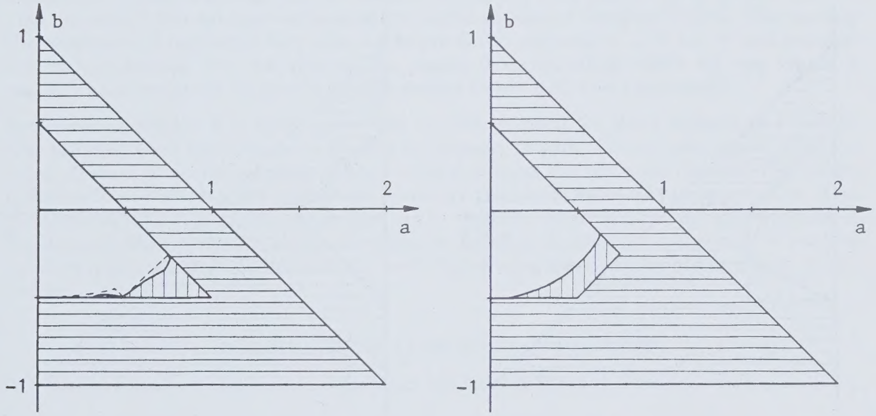

y
u
+
Σ
P
+
z^{-1}
x1
z^{-1}
x2
a
+
Q
Σ
+
b

Figure 3. Direct form digital filter with one quantizer



(a) Truncation quantizer, saturation or no overflow

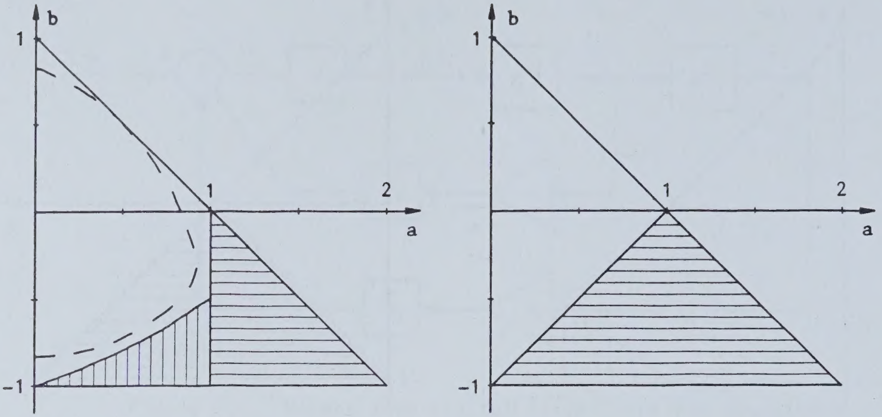
(b) Truncation quantizer, two's complement overflow



(c) Roundoff quantizer, saturation or no overflow

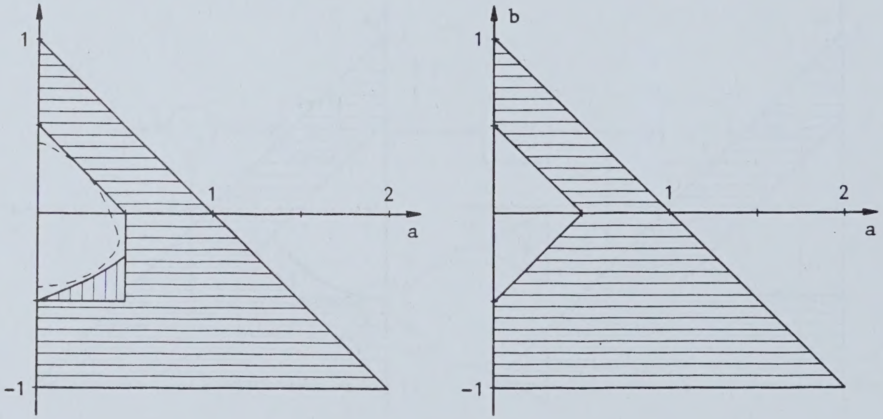
(d) Roundoff quantizer, two's complement overflow

Figure 4. Regions where the direct form filter with one quantizer and overflow is g.a.s. by the constructive algorithm



(a) Truncation quantizers, saturation or no overflow

(b) Truncation quantizers, two's complement overflow



(c) Roundoff quantizers, saturation or no overflow

(d) Roundoff quantizers, two's complement overflow

Figure 5. Regions where the direct form filter with two quantizers and overflow is g.a.s. by the constructive algorithm

FIR DIGITAL FILTERS WITH INCREMENTAL COEFFICIENT VALUES[*]

N. BENVENUTO
Bell laboratories
Holmdel, New Jersey

L. E. FRANKS and F. S. HILL, JR.
University of Massachusetts
Amherst, Massachusetts

ABSTRACT

Based on the idea of delta modulation, one approach to simplify the implementation complexity of FIR filters is to consider the filter structure consisting of a transversal filter with tap weights restricted to -1, 0, and +1 and cascaded with an accumulator. For this configuration some simple VLSI implementations are presented. Moreover, a novel algorithm for selecting the tap gains of the transversal filter such that the filter can meet given time- or frequency-domain specifications is illustrated.

INTRODUCTION

Recently, alternatives to the conventional (PCM) approach of representing digital signals have been introduced [1]-[4], [6]-[10]. In fact the traditional approach can often be too complex to implement due to multi-bit multipliers and adders. One approach to reduce the implementation complexity of the filter is to use a delta modulator (DM) to encode the input signal rather than a b-bit quantizer [1]-[4], [6]-[7]. The internal clocking rate may increase, but the resulting structure is simpler to implement. In this case, multipliers are eliminated, but still there may be many multi-bit adders. Thus, the idea of DM has been extended to the impulse response of the filter [8]-[10]. The resulting filter structure is a transversal filter with tap weights {d(n)} restricted to -1, 0, and +1 and cascaded with an accumulator. For this structure we present three realizations which are very simple to implement and mechanize in conjunction with modern trends of IC device technology.

The remaining problem is to design such filters, i.e., how to choose the {d(n)} sequence such that the filter can meet given specifications in the time or frequency domain. Using a mean square error criterion, dynamic programming methods are presented to select the values for the taps. The design procedure is very efficient, and comparison with other procedures shows that an improvement of up two orders of magnitude in the running time can be obtained. As we shall see the only drawback of the above structure is that for some applications the length of the transversal filter may be too long for some present devices. In this case the possibility of using more complex configurations is considered.

FILTERS USING DELTA MODULATION ENCODERS

The transfer function of a FIR filter of length N has the well-known D-transform representation

$$H^*(D) = \sum_{n=0}^{N-1} h(nT)\, D^n \tag{1}$$

where $h(0), h(T),...,h((N-1)T)$ comprise the unit pulse response of the filter. The frequency

---

[*] This work has been supported in part by the National Science Foundation under Grant ECS-8005688.

response is just $H^*(D)$ evaluated for $D = \exp(-j2\pi fT)$ and scaled by T, where T is the time delay between successive taps on the delay line.

$$H(f) = T \sum_{n=0}^{N-1} h(nT)\, e^{-j2\pi fnT} \tag{2}$$

In this case the output sequence $\{y(n)\}$ is given by the convolution of the tap weight sequence $\{h(nT)\}$ with input sampled sequence $\{x(nT)\}$:

$$y(nT) = \sum_{m=0}^{N-1} h(mT)\, x((n-m)T). \tag{3}$$

For a digital implementation the two sequences in (3) are often represented as 8-12 bit words. Thus it may be seen that calculation of each filtered sample requires N multiplications, $N-1$ additions, and the dynamic operation of time shifting the input sample sequence.

A completely different implementation results when the input signal is encoded into a binary sequence using a delta modulator rather than a conventional b-bit A/D converter [1]-[4], [6]-[7]. When linear delta modulation is used to encode the input signal [11], the operation required to reconstruct the original signal is a simple linear integration. Formally, let τ be the sampling interval and δ_x the step size of the delta modulator. Then samples of the reconstructed input are related by

$$\hat{x}(n\tau) = \hat{x}((n-1)\tau) + \delta_x u(n\tau) \tag{4}$$

where $u(n\tau)$ is the output from a one-bit quantizer:

$$u(n\tau) = sgn(x(n\tau) - \hat{x}((n-1)\tau)) \tag{5}$$

(sgn() is the signum function).

In order to obtain good performance (that is $\hat{x}(n\tau) \simeq x(n\tau)$ using DM a much higher sampling rate than the Nyquist rate is used [13]. We denote by K this oversampling factor, i.e.

$$K = \frac{1/\tau}{1/T} \tag{6}$$

where $1/T$ is the Nyquist sampling rate (twice the highest frequency component of the input signal) and $1/\tau$ is the sampling rate employed in the delta modulator.

Substituting (4) into (3) we obtain*

---

* Since the input step size δx represents only a gain scaling factor in the transfer function it has not been incorporated in (7).

$$y(n\tau) = y((n-1)\tau) + \sum_{m=0}^{N-1} h(mT) u((n-Km)\tau). \qquad (7)$$

Let $z(n\tau)$ represent the filtered version of $u(n\tau)$

$$z(n\tau) = \sum_{m=0}^{N-1} h(mT) \cdot u((n-Km)\tau). \qquad (8)$$

Then $y(n\tau)$ is simply the accumulated version of $z(n)$

$$y(n\tau) = y((n-1)\tau) + z(n\tau) \qquad (9)$$

as illustrated in Fig. 1.

In a digital implementation there are two major simplifications that make DM encoding attractive. The first is the simple digital implementation of the delta modulator to encode $x(t)$ [6]. The second simplification is in the implementation of the transversal filter [7].

FILTERS WITH INCREMENTAL COEFFICIENT VALUES

A further reduction in the complexity of the implementation is found by applying the concept of DM to the impulse response $h()$. Specifically we shall consider the case where the tap weights of the transversal filter, $d(n\tau)$, are restricted to -1, 0, or +1. The actual impulse response will then be the accumulated version of $\{d(n\tau)\}$, i.e.,

$$h(n\tau) = \sum_{i=1}^{n} d(i\tau). \qquad (10)$$

If the impulse response has the same DM sampling rate as the input signal, the resulting filter structure is a transversal filter with "ternary-valued" taps at every register stage, and cascaded with an integrator as illustrated in Fig. 2. This is in contrast to using one tap after every K delay, as is done in (8).

The improvement with respect to previous implementations derives from performing the summing operations with simple logic gates. For example an implementation for sampled data systems has been proposed by Bateman and Liu [8]. CCD devices are used to implement the transversal filter. Since the taps are -1, 0, or +1 the fabrication process is very simple. Moreover by using electronically-controlled gates at each tap the entire structure can be made programmable.

In digital filtering of binary inputs the basic filter structure can be implemented using three recirculating shift registers, one up-down counter, and one accumulator (see also [10]). For a transversal filter of length L, Fig. 3 shows the basic operations to obtain the output $v(n\tau)$ given by the convolution

$$v(n\tau) = \sum_{m=0}^{L-1} d((m+1)\tau) \cdot u((n-m)\tau) \qquad (11)$$

where $d(m\tau) \epsilon \{-1,0,+1\}$ and $u(m\tau) \epsilon \{-1,+1\}$.

This structure is actually a summing device of the ternary-valued sequence $d((m+1)\tau)\cdot u((n-m)\tau)$, $m=0,...L-1$. In Fig. 3, shift register SR1 contains the input sequence, SR2 and SR3 the impulse response bits and the following code has been used:

$$u'(m) = \begin{cases} 1 & \textit{if } u(m\tau) = +1 \\ 0 & \textit{if } u(m\tau) = -1 \end{cases}$$

$$d'(m) = \begin{cases} 1 & \textit{if } d(m\tau) = +1 \\ 0 & \text{if } d(m\tau) = -1 \textit{ or } d(m\tau) = 0 \end{cases} \tag{12}$$

$$d''(m) = \begin{cases} 0 & \text{if } d(m\tau) = 0 \\ 1 & \text{if } d(m\tau) = +1 \textit{ or } d(m\tau) = -1. \end{cases}$$

SR1 is shifted right and loaded with a new value of the input every τ seconds. All SR's are then clocked and recirculated at a rate L/τ. The up-down counter forms the running sum of the "products" $d((m+1)\tau)\cdot u((n-m)\tau)$ provided by the exclusive OR. The clock pulse CK is enabled only if the tap value $d((m+1)\tau)$ is not equal to zero. Configurations with a lower internal rate than L/τ can also be implemented at the expense of greater circuit complexity [10].

In order to reduce the internal clock rate a more attractive approach is to use a parallel implementation. Let L_o be the number of taps equal to zero, and let $\{b(i)\}$, $i = 1,...,L$, be determined as in (13).

$$b(m+1) = \begin{cases} 1 & \text{if } d((m+1)\tau) = u((n-m)\tau), \quad m = 0,1,...,L-1 \\ 0 & \textit{otherwise} \end{cases} \tag{13}$$

If $\ell(n)$ denotes the number of b's equal to 1, than the output $v(n\tau)$ is simply

$$v(n\tau) = 2\ell(n) - (L-L_o). \tag{14}$$

Therefore, $v(n\tau)$ can be obtained directly once we determine the number of 1's there are in the sequence $\{b(i)\}$, $i = 1,...,L$. Many methods may be used to implement this counter (also called "parallel counter"). For example, in IC technology a straightforward implementation of this approach would use a tally circuit with pass transistors [15]. Specifically, this circuit has L inputs $(b(1),...,b(L))$ and $L+1$ outputs $(C(0),C(1),...,C(L))$ and provides the i-th output to be high and all other outputs low if i of the inputs are high. (see also [16] Ch. 7). For the case $L = 7$, Figure 4 shows a schematic of a structure to obtain the binary representation of $\ell(n) = \ell_o + \ell_1 2 + \ell_2 2^2$ given u',d', and d'' (see (12)). The same symbology introduced in [15] has been used. The transistor layout has been shown to emphasize the high modularity of the implementation.

FILTER DESIGN

The following mean-square error objective functional proposed by Bateman and Liu [8] will be used

$$E = \sum_{n=-\infty}^{+\infty} [g(n\tau) - \delta h(n\tau)]^2. \tag{15}$$

In (15) $g()$ is the ideal impulse response, $\tau = T/K$ is the sampling period, δ is a scaling factor, and $\{h(n\tau)\}$ is the approximation sequence equal to the impulse response of the structure in Fig. 2.

In practical applications, the infinite series can be truncated to a finite number, say $N \cdot K$, of terms:

$$E = \sum_{n=n_o K+1}^{(n_o+N)K} [g(n\tau) - \delta h(n\tau)]^2 + E_R. \tag{16}$$

Let $M = N \cdot K$ and

$$\hat{g}(n\tau) = g((n+n_o K)\tau), \quad n = 1,2,...,M. \tag{17}$$

The above decomposition has a practical interest if a fast method for optimizing

$$E_M = \sum_{n=1}^{M} [\hat{g}(n\tau) - \delta h(n\tau)]^2 \tag{18}$$

is available, and if the truncation error E_R is negligible with respect to E.

An optimum algorithm based on dynamic programming methods [12] in conjunction with the functional (19) has been used to select the values for the taps $\{d(n)\}$.

Unfortunately, the algorithm works with a fixed value of the step size δ. To minimize E_M with respect to δ the same procedure suggested in [8] was used.

To test the optimization procedure, the above algorithm was implemented on a CDC Cyber-175 for an underdamped second order filter: i.e. using a target impulse response given by

$$g(t) = \frac{w_n}{\sqrt{1-\xi^2}} e^{-\xi t} \sin\left(\sqrt{1-\xi^2} w_n t\right) \tag{19}$$

with $w_n=1$ and $\xi=0.8$, and various values of τ.

For comparison purposes we also implemented the algorithm proposed by Bateman and Liu in [8], where for a given step size δ an iterative perturbation method was presented to find the optimum sequence $\{d(p)\}$.

Table 1 shows the number of steps used ($\#\delta$) and the total computer execution time (in seconds) to reach the minimum of E_M for various values of τ. The comparison reveals a considerable improvement (in some cases two orders of magnitude) using the dynamic programming method over the perturbation method.

EXAMPLE OF APPLICATION

To give some insight into the behavior of the filter structure shown in Fig. 2, we present a particular example applying the design procedure.

As an ideal transfer function we select a normalized "double-jump" rolloff function (DJ) [5], i.e.

$$G(f) = \begin{cases} 1 \quad , & 0 < f < (1-\gamma)/2 \\ 1-f \ , & (1-\gamma)/2 \leqslant f \leqslant (1+\gamma)/2 \\ 0 \quad , & f > (1+\gamma)/2. \end{cases} \tag{20}$$

The corresponding ideal impulse response is given by:

$$g(t) = (1-\gamma)\ sinc(1-\gamma)t + sinc\gamma t\ \cos\pi t + \tag{21}$$

$$-\ \frac{\pi\gamma t\ \cos\pi\gamma t - \sin\pi\gamma t}{\pi^2 t^2}\ \sin\pi t.$$

For a rolloff factor $\gamma = 0.4$ we choose $T = 0.5$ and $N = 100$.

Figures 5 and 6 show results of the optimum filter design using this structure for different values of K. With the optimum value of δ, the length L of the sequence $\{d(p)\}$ (without the zero "tails") is reported. In the figures solid lines represent the ideal case.

It is evident that we can increase the accuracy of the DM-like form of $g()$ by increasing the oversampling parameter K. On the other hand, referring to Fig. 6, we may need an excessively large value of K to obtain an acceptable error with the result that the number of taps L will be unacceptably large.

In order to limit the length of these filters, in [14] we examine extensions of the delta modulation concept and introduce other types of structures.

REFERENCES

[1] G. B. Lockhart, "Digital encoding and filtering using delta modulation," *The Radio and Electronic Engineer*, vol. 42, pp. 547-551, Dec. 1972.

[2] G. B. Lockhart, "Binary transversal filters with quantized coefficients," *Electronic Letter*, vol. 7, pp. 305-307, June 1971.

[3] L. E. Franks, "Hybrid implementation of sampled-data filters," IEEE Int. Symp. Circuits and Systems, San Francisco, April 1974.

[4] L. E. Franks, "Hybrid implementation of digital filters," European Conference on Circuits Theory and Design, London, July 1974.

[5] L. E. Franks, "Further results on Nyquist's problem in pulse transmission," *IEEE Trans*, COM-16, pp. 337-340, April 1968.

[6] L. E. Franks, "Signal processing structures based on digital incremental computing techniques," Int. Conf. on Digital Signal Processing, Florence (Italy), Sept. 1981.

[7] A. Peled and B. Liu, "A new approach to the realization of nonrecursive digital filter," *IEEE Trans*, AU-21, pp. 477-484, Dec. 1973.

[8] M. R. Bateman and B. Liu, "An approach to programmable CTD filters using coefficients 0, +1, and -1," *IEEE Trans.*, CAS-27, pp. 451-456, June 1980.

[9] G. B. Lockhart, "Design and application of binary sequences with optimized lowpass properties," *Computers and Digital Techniques*, vol. 1, pp. 197-204, Oct. 1978.

[10] G. B. Lockhart and S. P. Babary, "Binary transversal filters using recirculating shift registers," *The Radio and Electronic Engineer*, vol. 43, pp. 224-226, March 1973.

[11] H. R. Schindler, "Delta modulation," *IEEE Spectrum*, vol. 7, pp. 69-78, Oct. 1970.

[12] H. M. Wagner, *Principles of operations research*, II Ed., Englewood Cliffs, N. J.: Prentice-Hall, 1975.

[13] N. S. Jayant, "Digital coding of speech waveforms: PCM, DPCM, and DM quantizer," *Proc. IEEE*, vol. 62, pp. 611-632, May 1974.

[14] N. Benvenuto, L. E. Franks, and F. S. Hill, Jr., "Realization of finite impulse response filters using coefficients +1, 0, and -1," submitted to IEEE Trans. on Comm., Sept. 1982.

[15] C. Mead and L. Conway, *Introduction to VLSI systems*, Addison-Wesley Publisher, 1980.

[16] S. H. Caldwell, *Switching circuits and logic design*, New York, N.Y.: John Wiley Sons, Inc., 1958.

Table 1. Computation time (sec.).

M	τ	$\#\delta$	Perturbation Algorithm [8]	Dynamic Progra. Algorithm
10	0.50	8	0.229	0.075
20	0.25	8	1.164	0.101
50	0.10	19	37.595	0.511
100	0.05	9	---	0.667

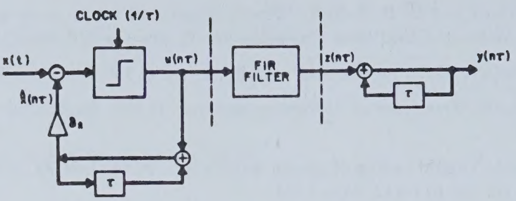

CLOCK (1/τ)
x(t)
-
x̂(nτ)
a₁
u(nτ)
FIR FILTER
z(nτ)
+
τ
y(nτ)
+
τ

FIGURE 1 FILTERING USING DM ENCODING

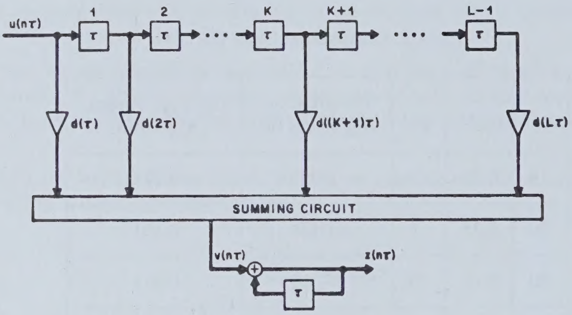

u(nτ)
1
τ
2
τ
K
τ
K+1
τ
L-1
τ
d(τ)
d(2τ)
d((K+1)τ)
d(Lτ)
SUMMING CIRCUIT
v(nτ)
+
τ
z(nτ)

FIGURE 2 FIR FILTER WITH SIMPLIFIED IMPLEMENTATION

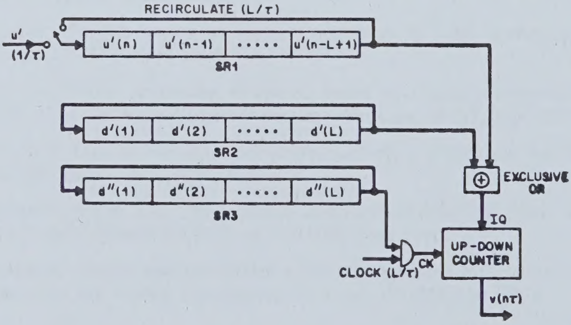

RECIRCULATE (L/τ)
u'
(1/τ)
u'(n)
u'(n-1)
.....
u'(n-L+1)
SR1
d'(1)
d'(2)
.....
d'(L)
SR2
d''(1)
d''(2)
.....
d''(L)
SR3
⊕
EXCLUSIVE OR
IQ
UP-DOWN COUNTER
CK
CLOCK (L/τ)
v(nτ)

FIGURE 3 SERIAL IMPLEMENTATION OF BTF WITH "TERNARY VALUED" COEFFICIENTS

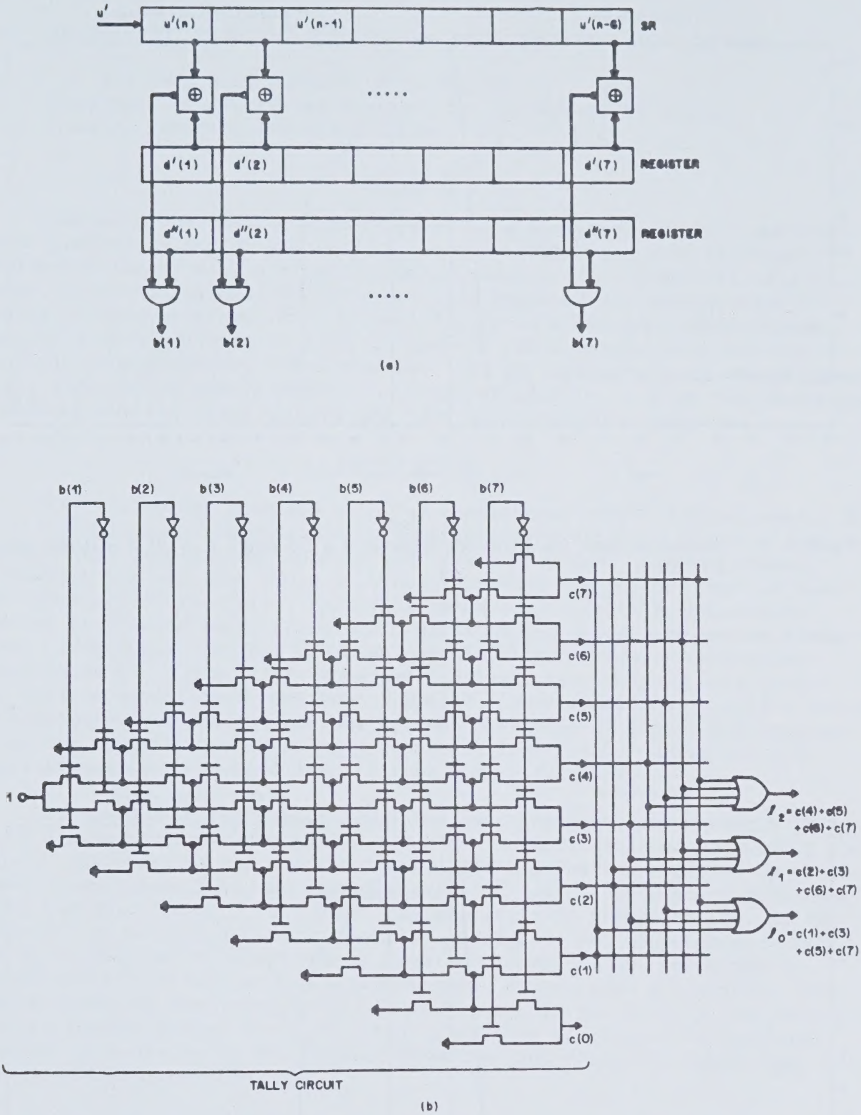

u'
u'(n)
u'(n-1)
u'(n-6)
SR
d'(1)
d'(2)
d'(7)
REGISTER
d''(1)
d''(2)
d''(7)
REGISTER
b(1)
b(2)
b(7)
(a)
b(1)
b(2)
b(3)
b(4)
b(5)
b(6)
b(7)
1
c(7)
c(6)
c(5)
c(4)
c(3)
c(2)
c(1)
c(0)
f_2 = c(4) + c(5) + c(6) + c(7)
f_1 = c(2) + c(3) + c(6) + c(7)
f_0 = c(1) + c(3) + c(5) + c(7)
TALLY CIRCUIT
(b)

FIGURE 4 PARALLEL IMPLEMENTATION OF BTF WITH "TERNARY VALUED" COEFFICIENTS

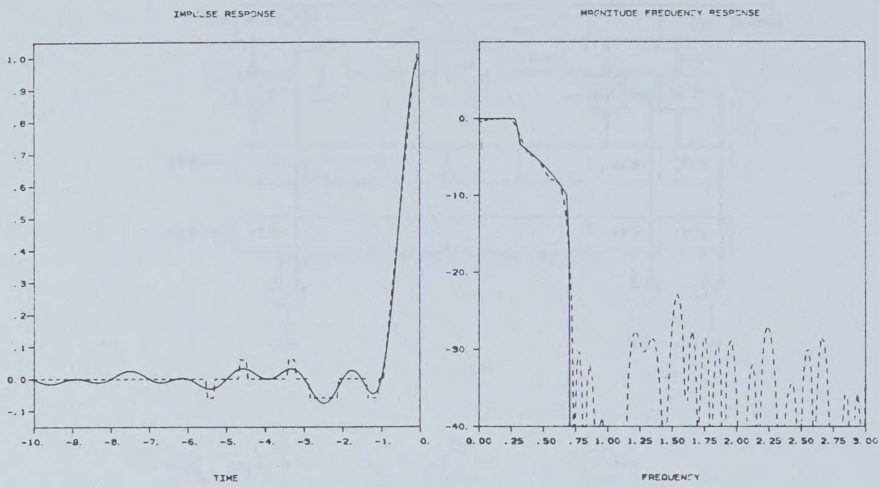


Figure 5 Impulse response and frequency response of a DJ filter. $K = 10$, $\delta = 0.060$, and $L = 221$.

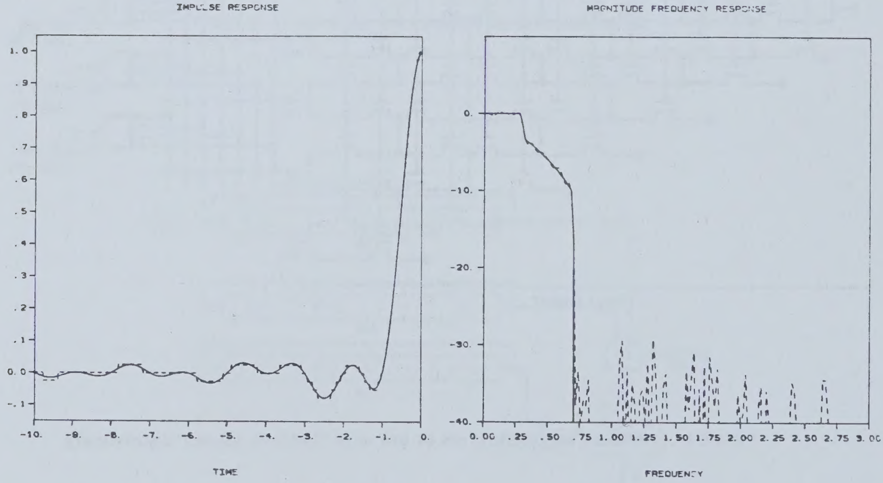


Figure 6 Impulse response and frequency response of a DJ filter. $K = 26$, $\delta = 0.026$, and $L = 1317$.

VLSI IMPLEMENTATION OF RNS MODULES AND THEIR PROPOSED APPLICATIONS

M.A. BAYOUMI, G.A. JULLIEN, W.C. MILLER
Department of Electrical Engineering, University of Windsor
Windsor, Ontario, N9B 3P4, Canada

ABSTRACT

Combining the Residue Number System as a computational tool and VLSI as a fabrication medium promises to provide modular and cost efficient implementation of many digital signal processing algorithms. In this paper, a one-dimensional systolic array is proposed for implementing a finite impulse response (FIR) digital filter in a residue number system. The building block unit is a multi look-up table module which has two possible configurations. The area and time of this module can be estimated using a developed memory model. A numerical example is given for designing a modular RNS FIR based on area and time as performance measures.

1. INTRODUCTION

VLSI technology promises to offer substantial computational power. In this technology, it is possible to construct sophisticated circuits that perform very complex functions in very small packages. Digital signal processing applications stand to benefit enormously, perhaps more so than any other application area, from VLSI technology. In this environment special purpose hardware chips for digital signal processing become attractive. Tremendous progress has been made in recent years on developing these chips [1]. Many algorithms and architectures that are well suited for VLSI implementation have been proposed. More importantly, useful design methodologies for special-purpose systems have been evolved. For example, using the systolic approach, many important functional modules for DSP applications can be mapped into silicon cost-effectively.

The Residue Number System (RNS) has been proven by many authors [2-4] to be a successful implementation for high DSP applications. Combining the modular structure of RNS and the technological advantages offered by VLSI leads to enhancement in RNS implementation in speed, cost, power dissipation, and chip density. The VLSI implementation is promising [5] as RNS supports the main VLSI design features; modularity and concurrency. RNS has a parallel nature where the arithmetic operations are performed independently for each module which supports distributed processing. The memory intensive architecture [4] is very suitable for VLSI, as the memory modules achieve higher level of integration than any other computational element; this is due to the regular geometry, and straight, simple and regular interconnections.

In this paper a standard module has been developed to be the building block unit of 1-D systolic array based on RNS for performing FIR processing. The area and time delay of this module can be evaluated using a memory model developed by the authors. Two different structures have been proposed depending on different configurations of the proposed module. Finally, an analysis for area and time has been developed for a FIR RNS filter.

2. RNS REVIEW

The Residue Number System (RNS) is an integer number system defined over a set of relatively prime integers [6], $\{m_1, m_2, \ldots, m_L\}$ called moduli.

The product of the moduli $M = \prod_{i=1}^{L} m_i$ constitutes the dynamic range of the RNS. Every integer $X \varepsilon \left[-\frac{M}{2}, \frac{M}{2} - 1 \right]$ has residue representation

$$X = \left(x_1, x_2, \ldots, x_L\right) \tag{1}$$

where

$$x_i = \begin{cases} X \bmod m_i \ \left(|X|_{m_i}\right) & \text{if } X > 0 \\ \left(M - |X|_{m_i}\right) \bmod m_i & \text{if } X < 0 \end{cases}$$

If $X = \left(x_1, x_2, \ldots, x_L\right)$ and $Y = \left(y_1, y_2, \ldots, y_L\right)$ then $Z = X \circ Y = \left(z_1, z_2, \ldots, z_L\right)$, where $z_i = \left|x_i \circ y_i\right|_{m_i}$ and '$\circ$' represents addition, subtraction or multiplication. The attractive property of the RNS is that the z_i's can be computed concurrently without carry propagation. Typically $\left|x_i \circ y_i\right|_{m_i}$ is computed via high speed table look-up, though other implementation have also been proposed [10].

3. MEMORY MODEL

A memory model has been developed to estimate the area and access time of a memory module required for storing a certain look-up table. This model is flexible having these features:

(a) the word width is $\lceil \log_2 (m-1) \rceil$ where m is the modulus.

(b) the storage array is efficient, it can be divided into multi-subarrays, its dimensions are not necessarily a power of two.

(c) the number of decoders outputs is m^2, this leads to substantial saving in the required area.

An algorithm has been developed to obtain all the viable memory layout options for a given modulus as shown in the table below for m = 11.

Length	Width	Area	Time	Area Time Product
126.1	27.1	3417.59	121	413527.8
68.2	23.8	1624.89	61	99118.29
50.3	26.8	1347.61	41	55252.14
40.1	28.1	1135.85	31	34211.42
36.7	31.9	1168.33	25	29208.21
32.8	35.7	1171.57	24	28117.78
30	39.6	1186.34	28	33217.15
28	41.4	1163.54	32	37233.32

where Length and Width are in unit length, Area in unit area and Time in unit time.

An algorithm has been developed to select the most efficient layout option according to a certain cost function determined by the designer [7].

4. A STANDARD MODULE

In the VLSI environment, the designer has control over the chip function, geometry and topology, so some modules can be built in which several independent look-up tables are stored in the same chip with appropriate interconnections. In this paper a module having 2 look-up tables has been proposed as shown in Fig. 1. It is called the CAS-module. Each look-up table can represent an RNS mathematical operation (+ , − , or $*$) between two variables with an added capability of an operation with a constant on each of the variables. As an example, Fig. 1 shows the CAS-module programmed to perform

$$Y = \left| \left|X_1\right|_{m_i} * \left|X_2\right|_{m_i} + \left|\alpha X_3\right|_{m_i} \right|_{m_i}$$

To have a flexible module suitable for most of DSP applications, a control pin has been added to determine different possible paths for the input signals as shown in Fig. 2. This module has two possible configurations as follows:

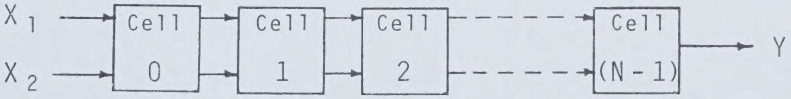
$C = 1$ configuration I or CAS. I

$C = 0$ configuration II or CAS.II

Figure 3 shows both configurations. Using the developed memory model and the procedures proposed in [7], one can get an estimate for area and time required for building this module for a certain set of moduli.

5. SYSTOLIC ARCHITECTURES

The high degree of concurrency, which can be provided through VLSI technology has been exploited in some special purpose machines called the systolic array devices designed by Kung [8]. A detailed description of the basic principles of systolic architectures together with several particular designs is given in the above reference and will not be repeated here. Typically, a systolic array consists of a number of identical processing elements of simple computational complexity. These elements are locally connected by a regular interconnection pattern. Data entering the systolic array is processed by a set of the elements before being output, thus allowing a high degree of concurrency. The essential features of systolic arrays are their simplicity, regularity, and modularity. Performance-wise, these characteristics ensure high level of pipelining and multiprocessing hence providing massive parallelism. Their modularity permits the decomposition of the system's architecture into building blocks which can be used repetitively with simple interfaces. Systolic arrays can be arranged either in one or two-dimensional arrangements. In this paper only one-dimensional arrays are considered, as shown:



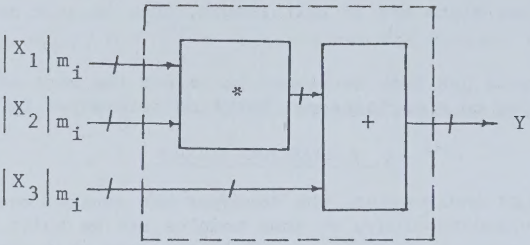
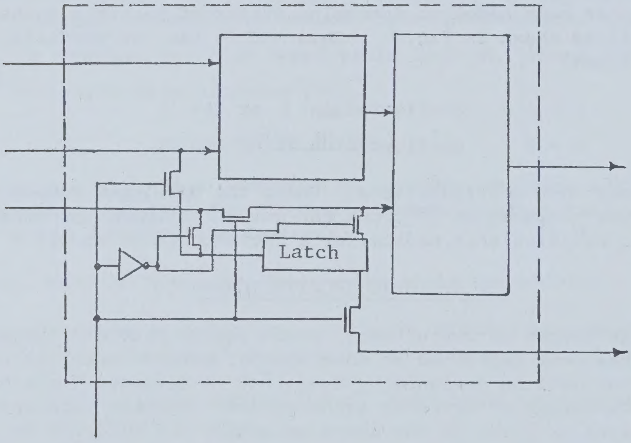

$|X_1|_{m_i}$
$|X_2|_{m_i}$
$|X_3|_{m_i}$
*
+
Y

Fig. 1 The CAS Module



C = 1 configuration I (CAS-I)
0 configuration II (CAS-II)

Fig. 2 A General CAS Module

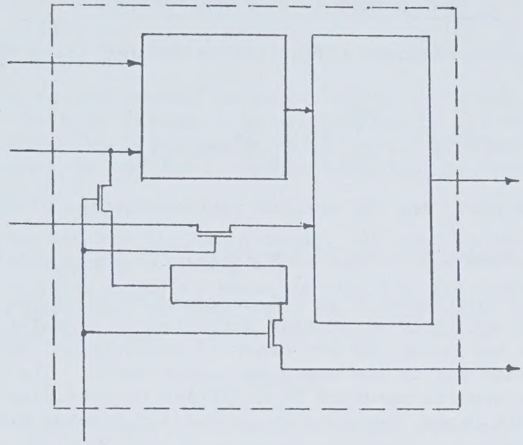


Fig. 3a CAS-I Configuration

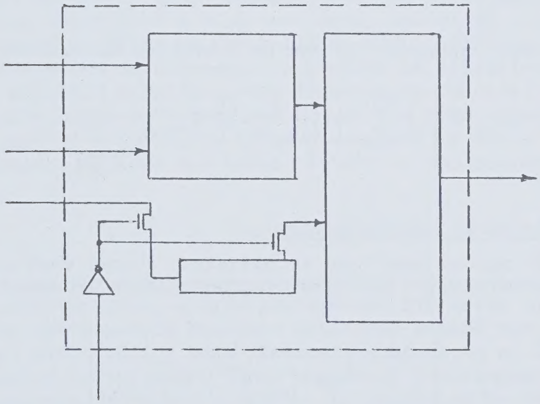


Fig. 3b CAS-II Configuration

6. FINITE IMPULSE RESPONSE FILTERS

A finite impulse response filter can be defined using RNS as:

$$y(k)_j = \left| \sum_{i=0}^{N-1} \left|\alpha_i\right|_{m_j} \left|x_{k-i}\right|_{m_j} \right|_{m_j} \tag{2}$$

where $\left|x_{k-i}\right|_{m_j}$ and $y(k)_j$ are the residue representations of the j-th residue of the input and output signals of the filter and $\left|\alpha_i\right|_{m_j}$, $i = 0, 1, 2, \ldots, N-1$ are the filter coefficients in RNS representation of the j-th residue. These indices are naturally associated with time in a real-time environment. Equation (2) does not prescribe any order of computation, so, each value of $y(k)_j$ can be carried out in any order or concurrently. The filter can be implemented using several hardware realizations by rewriting (2) in different forms. In this paper, our concern is the distributed processing realization.

The mathematical vehicle in showing the correspondence between the mathematical forms and the hardware realization. In z-transform the time delay is expressed as follows:

$$Z^{-1} \left|X(z)\right|_{m_j} \longleftarrow \left|X(i-1)\right|_{m_j}$$

$$Z^{-\ell} \left|X(z)\right|_{m_j} \longleftarrow \left|X(i-\ell)\right|_{m_j}$$

In transform domain, equation (2) can be rewritten as follows:

$$Y(z)_j = \left| \left(\sum_{i=0}^{N-1} \left|\alpha_i\right|_{m_j} Z^{-i} \right) \left|X(z)\right|_{m_j} \right|_{m_j} \tag{3}$$

6.1. 1-D Systolic Array Implementation

Equation (3) can be rewritten in different forms, each corresponds to different implementation . The first implementation is based on the following form:

$$Y(z)_j = \left| \left(\sum_{i=0}^{N-1} \left|X(z)\right|_{m_j} Z^{-1} \right) * \left|\alpha_i\right|_{m_j} \right|_{m_j} \tag{4}$$

Equation (4) can be implemented as shown in Fig. 4, which is a modular implementation. It is a linear array of the same module. In this scheme, the index i indicates the module number (space) and the index K expresses time. The building block unit is the CAS-I module proposed in section 4 programmed as shown in Fig. 5.

Another possible implementation of FIR using 1-D array is based on rewriting (3) as follows:

$$Y(z)_j = \left| \sum_{i=0}^{N-1} \left(\left| X(z) \right|_{m_j} * \left| \alpha_i \right|_{m_j} \right) z^{-i} \right|_{m_j} \qquad (5)$$

Equation (5) can be implemented as shown in Fig. 6, which is a linear modular array. In this scheme, X is broadcasted to all modules. For high order filters, this causes a time delay [9] which is undesirable. The building block unit is the CAS-II module developed in section 4 programmed as shown in Fig. 7.

To design an RNS FIR filter, a moduli set must be selected which provides enough dynamic range. Using either one of the two schemes proposed above, the overall FIR array is shown in Fig. 8. The computation for each modulus is performed in parallel. The scalar unit is used when it is required to adjust the computation range within the used dynamic range.

7. EXAMPLE

To design an RNS based FIR filter, a moduli set must be selected which provides enough dynamic range. Consider the 24th-order lowpass filter described in [11]. The filter coefficients can be represented by 10-bit to get a high precision. Assuming that the input data is quantized to 8 bits and the filter maintains minimal rounding error, a reasonable dynamic range can be (16-17) bits, roughly. Selecting a moduli set of $\{14, 13, 11, 9\}$ provides a dynamic range of 15 bits which still is adequate for most practical situations.

For this specific example, 4 arrays are constructed each for one modulo computation. The size of the building block module differs according to the used modules. As an example, for the array module 11, an efficient module is of 61.9 x 40.1 unit area, and of 55 unit time (section 3), Fig. 9. The time delay of the building module differs according to the used modulus which is considered a problem in realizing pipelining. There are two possible solutions, the first is to consider the largest time delay as the basic reference and adjust the time delay of the other modules. The second solution is to have a buffer at the end of each array (before the scaler) to have all input signals to the scaler at the same time.

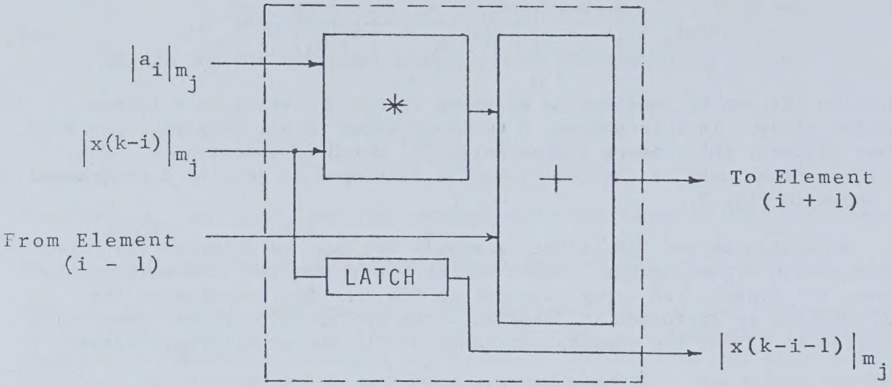
8. CONCLUSIONS

By exploiting the regularity, parallelism and pipelining inherent in the systolic architectures, a high performance FIR array has been proposed. A multi look-up table module has been developed having two configurations using a control pin. Adding this flexibility leads to an increase in area but the time delay is the same. This module is programmable. The implementation parameters (area and time) can be estimated by using the developed memory model. The optimal layout is determined by a procedure developed in [7]. Two structures have been proposed for FIR filter implementation, both of them are 1-D systolic array.

REFERENCES

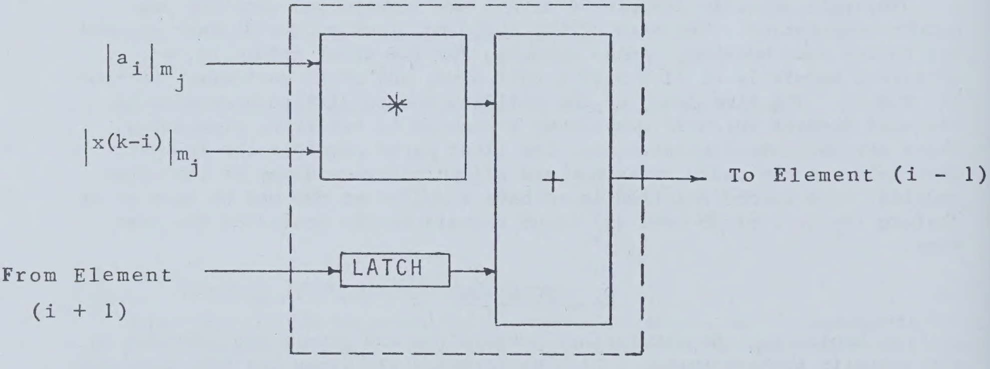
[1] M.J. Fosters and H.T. Kung, "The Design of Special-Purpose VLSI Chips", Computer Magazine, Vol.13, pp.26-40, 1980.

[2] W.K. Jenkins and B.J. Leon, "The Use of Residue Number Systems in the Design of Finite Impulse Response Digital Filters", IEEE Trans.



$|a_i|_{m_j}$
$|x(k-i)|_{m_j}$
From Element (i - 1)
*
+
LATCH
To Element (i + 1)
$|x(k-i-1)|_{m_j}$

Fig. 5 CAS-I as Element i in the Array of Fig. 4



$|a_i|_{m_j}$
$|x(k-i)|_{m_j}$
From Element (i + 1)
*
+
LATCH
To Element (i - 1)

Fig. 7 CAS-II As Element i In The Array of Fig. 6

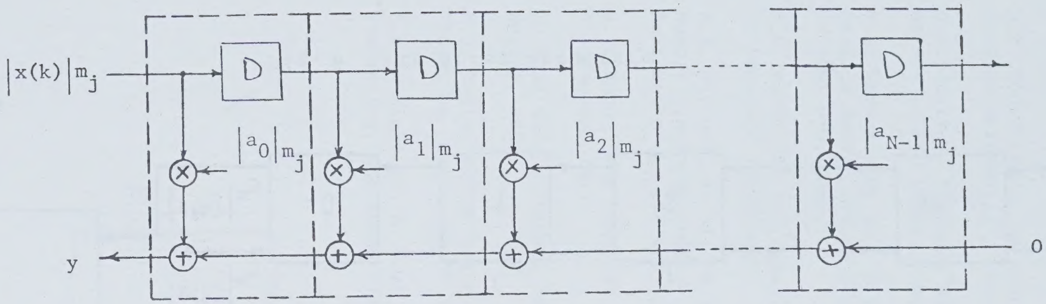

$|x(k)|_{m_j}$
D
D
D
D
$|a_0|_{m_j}$
$|a_1|_{m_j}$
$|a_2|_{m_j}$
$|a_{N-1}|_{m_j}$
y
0

Fig. 4 1-D Systolic Array FIR

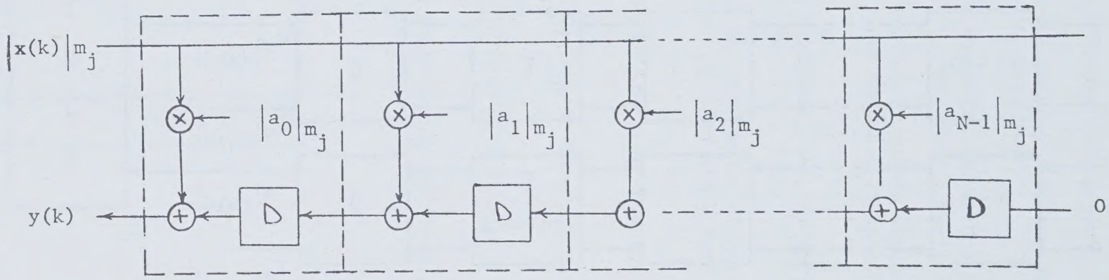
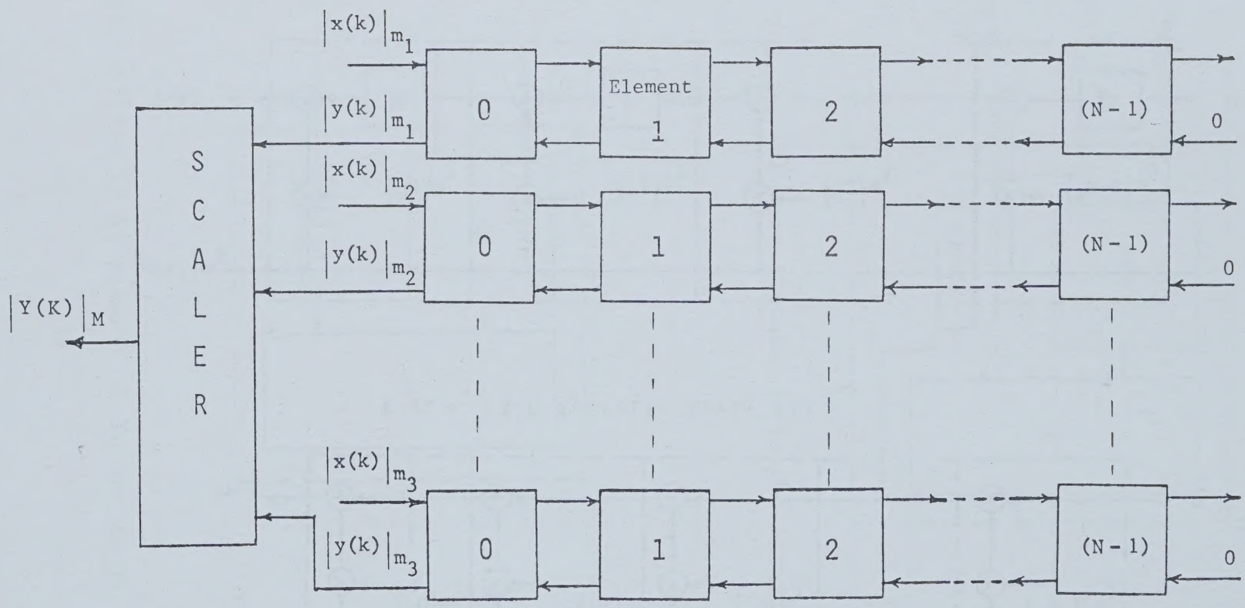

$|x(k)|_{m_j}$
$|a_0|_{m_j}$
$|a_1|_{m_j}$
$|a_2|_{m_j}$
$|a_{N-1}|_{m_j}$
D
D
D
y(k)
0

Fig. 6 1-D Systolic Array FIR



$|x(k)|_{m_1}$
$|y(k)|_{m_1}$
$|x(k)|_{m_2}$
$|y(k)|_{m_2}$
$|x(k)|_{m_3}$
$|y(k)|_{m_3}$
0
Element 1
2
(N - 1)
0
1
2
(N - 1)
0
1
2
(N - 1)
0
0
0
S C A L E R
$|Y(K)|_M$

Fig. 8 FIR Array Structure

Circuits and Systems, Vol. CAS-24, pp.191-201, April 1977.

[3] M.A. Soderstrand, "A High Speed Low-Cost Recursive Filter Using Residue Arithmetic", Proc. IEEE, Vol.65, July 1977.

[4] G.A. Jullien, "Residue Number Scaling and Other Operations Using ROM Arrays", IEEE Trans. Comput., Vol. C-27, pp.325-337, April 1978.

[5] M.A. Bayoumi, et al, "A VLSI Implementation of Memory Intensive Residue Number System Architecture", 20th Annual Allerton Conf., October 6-8, 1982.

[6] N.S. Szabo and R.I. Tanaka, "Residue Arithmetic and Its Applications to Computer Technology", McGraw-Hill, New York, 1967.

[7] M.A. Bayoumi, et al, "Models for VLSI Implementation of Residue Number System Arithmetic Modules", Proc. of IEEE Sixth Symp. on Comp. Arithmetic, pp.174-182, June 1983.

[8] H.T. Kung, "Why Systolic Architectures ?" Computer Magazine 15(1):37-46, January 1983.

[9] C. Mead and L. Conway, "Introduction to VLSI Systems", Addison-Wesley, 1980.

[10] F.J. Taylor and H. Huang, "An Autoscale Residue Multiplier", IEEE Trans. on Compt., April 1981.

[11] L.R. Rabiner and B. Gold, Theory and Applications of Digital Signal Processing, Englewood Cliffs, N.J., Prentice-Hall 1975.

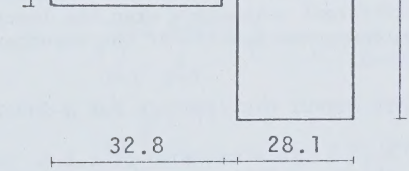


Fig. 9 An outline of the CAS module modulo 11

SOME STATISTICAL PROPERTIES OF GENERALIZED MEDIAN FILTERS*

YONG HOON LEE and SALEEM A. KASSAM
Department of Systems Engineering
Moore School of Electrical Engineering
University of Pennsylvania
Philadelphia, Pennsylvania 19104

ABSTRACT

The uni- and bivariate distribution functions of an M-smoothed sequence of i.i.d. random variables are derived, and those of the median and the running mean filters are obtained as specific cases. The bivariate distribution function is then used to determine autocorrelations and spectral densities of M-smoother outputs with different parameters when the input random variables have finite alphabets. The results exhibit a low-pass filter characteristic for the M-smoothers, the running mean filter and the median filter showing the best and the worst low-pass characteristics, respectively, for fixed window and input alphabet sizes.

I. INTRODUCTION

Median filters have received considerable attention as useful alternatives to linear filters in applications such as speech and image processing [1-4]. One of their interesting characteristics is that they can preserve sharp discontinuities in a signal while removing spiky noises. In many signal processing applications, however, median filtering may fail to provide sufficient smoothing of undesirable noise-like components because of its extreme nonlinear characteristic [2]. Recently, nonlinear smoothers called the L-smoother (or order statistic filter [5]) and the M-smoother [6] were introduced as generalizations of median filters. The smoothing behaviour of both L- and M-smoothers can be controlled through parameters that can be set, so that the performances of the smoothers are in between those of median filters and of linear filters. Therefore, in some situations L- or M-smoothers may be preferred to median or linear filters.

In this paper we will investigate some statistical properties of M-smoothed outputs when the input is assumed to be a sequence of i.i.d. random variables. In particular, the output uni- and bivariate distribution functions are derived, and those of the median and the running mean filters are obtained as specific cases. The bivariate distribution functions are used to numerically determine the autocorrelation functions for M-smoothers with different parameters when the input signals are discrete valued. Then the power spectra of the smoothers for these cases are obtained and analyzed.

II. UNIVARIATE OUTPUT DISTRIBUTION FOR M-SMOOTHER

Suppose that $X_1, X_2, \ldots$ is a sequence of i.i.d. random variables with distribution function $F_x(x)$ and density function $f_x(x)$. An M-smoother of window size 2N+1 produces an output sequence Y_n for which

$$\sum_{i=n-N}^{n+N} \phi(X_i - Y_n) = 0$$

---

*This research is supported by the Air Force Office of Scientific Research under Grant AFOSR 82-0022.

for $n \ge N+1$. Here ϕ is in general some odd function. A case of particular interest is the standard type M-smoother (STMS), where ϕ is defined by

$$\phi(x) = \begin{cases} p & \text{if } x > p \\ x & \text{if } |x| \le p \\ -p & \text{otherwise} \end{cases} \tag{1}$$

with p a positive number. The STMS becomes the median filter or the running mean filter with window size $2N+1$ as p approaches 0 or ∞, respectively. It is proved in [6] that the output of an STMS is determined uniquely. In addition it can readily be seen that the output process of the STMS is strictly stationary. Thus the univariate output distribution for an STMS is given by

$$\begin{aligned} P(Y_{N+1} \le y) &= P\Big(\sum_{i=1}^{2N+1} \phi(X_i - y) \le 0\Big) \\ &\triangleq P\Big(\sum_{i=1}^{2N+1} \phi(Z_i) \le 0\Big) \\ &= \sum_{k=0}^{2N+1} \sum_{\ell=0}^{2N+1-k} \binom{2N+1}{k}\binom{2N+1-k}{\ell} \\ &\quad \cdot P\Big(\sum_{i=1}^{2N+1} \phi(Z_i) \le 0 \,\Big|\, |Z_i| \le p \text{ for } \\ &\qquad i = 1, \ldots, k,\ Z_i > p \text{ for } i = k+1, \ldots, k+\ell, \\ &\qquad Z_i < -p \text{ for } i = k+\ell+1, \ldots, 2N+1\Big) \\ &\quad \cdot [F_Z(p) - F_Z(-p)]^k \cdot [1 - F_Z(p)]^\ell \cdot [F_Z(-p)]^{2N+1-(k+\ell)} \\ &= \sum_{\ell=0}^{N} \binom{2N+1}{\ell} [1 - F_Z(p)]^\ell [F_Z(-p)]^{2N+1-\ell} \\ &\quad + \sum_{k=1}^{2N+1} \sum_{\ell=0}^{2N+1-k} \binom{2N+1}{k}\binom{2N+1-k}{\ell} \\ &\quad \cdot P\Big(\sum_{i=1}^{k} Z_i \le c,\ |Z_i| \le p \text{ for } i = 1, \ldots, k\Big) \\ &\quad \cdot [1 - F_Z(p)]^\ell [F_Z(-p)]^{2N+1-(k+\ell)} \end{aligned} \tag{2}$$

where $F_Z(z) = F_x(z+y)$ and $c = (2N+1-k-2\ell)p$. The first term after the last equality in (2) corresponds to $k=0$. Let

$$P_J = P\Big(\sum_{i=1}^{k} Z_i \le c,\ |Z_i| \le p \text{ for } i = 1, \ldots, k\Big)$$

and

$$P_A = P(|Z_i| \le p \text{ for } i = 1, \ldots, k)$$

In order to obtain P_J define

$$Q_k(j_k, j_{k-1}, \ldots, j_2, j_1) = \int_{b_1}^{q_1} \int_{b_2}^{q_2} \cdots \int_{b_k}^{q_k} \prod_{j=1}^{k} f_Z(z_j)\,dz_k \ldots dz_1 \tag{3}$$

where j_n's are either 0 or 1, $q_n = c - [\sum_{j=1}^{n-1} z_j - (k-n)p]$ and

$$b_n = \begin{cases} -p, & j_n = 0 \\ p, & j_n = 1 \end{cases}$$

for $n = 1, 2, \ldots, k$. We assume that the sum is zero when the upper limit is smaller than the lower limit. This will be assumed for integrals also, throughout the paper. Let

$$S_k(B) = Q_k(j_k, j_{k-1}, \ldots, j_2, j_1)$$

where $B = \sum_{i=1}^{k} j_i \cdot 2^{i-1}$, i.e. $(j_k, j_{k-1}, \ldots, j_1)$ is the binary expansion of B.

Then

$$P_J = \begin{cases} 0 & \text{when } c \le -kp \\ P_k(h) & \text{when } -(k-2h)p < c \le -(k-2h-2)p \\ P_A & \text{when } c > kp \end{cases} \tag{4}$$

where $0 \le h \le k-1$,

$$P_k(h) = S_k(0) - \sum_{i=0}^{k-1} [S_k(2^i) - \sum_{B \varepsilon E(i,h)} S_k(B)] \tag{5}$$

and $E(i,h)$ is the set of all integers j where $2^i < j < 2^{i+1}$ with the binary expansion of j containing at most a number h of 1's. The output univariate distribution is obtained by substituting (4) in (2). The output univariate distributions of two specific STMS's, the median and the running mean filters, will be considered in Section IV.

III. BIVARIATE OUTPUT DISTRIBUTION FOR M-SMOOTHER

Consider two output samples Y_n and Y_m for which $|m-n| < 2N+1$. For $|m-n| \ge 2N+1$, Y_n and Y_m are independent. Now for $|m-n| < 2N+1$, the number of input samples contributing to both Y_m and Y_n is $L = 2N + 1 - |m-n|$. Let us define for some fixed m and n, the sets (as in [7])

M_μ = {samples of the sequence generating Y_μ that are not in the overlap}

Q = {samples of the overlap}

where μ = m or n. Since Q contains L elements, M_m and M_n have 2N+1-L elements each. To consider the elements of Q we divide the real line into five open and closed intervals $I_1, \ldots, I_5$ for $o < p < \infty$:

$$I_1 = (-\infty, \min(y_m-p, y_{n-p}))$$

$$I_2 = \Big[\ [\min(y_m-p, y_n-p), \min(y_m+p, y_n+p)] \quad \text{when (i)}$$

$$\bigg\lfloor \; [\min(y_m-p,\ y_n-p),\ \max(y_m-p,\ y_n-p)] \quad \text{when (ii)}$$

$$I_3 = \begin{cases} (\min(y_m+p, y_n+p),\ \max(y_m-p,\ y_n-p)) & \text{when (i)} \\ (\max(y_m-p, y_n-p),\ \min(y_m+p,\ y_n+p)) & \text{when (ii)} \end{cases}$$

$$I_4 = \begin{cases} [\max(y_m-p,\ y_n-p),\ \max(y_m+p,\ y_n+p)] & \text{when (i)} \\ [\min(y_m+p,\ y_n+p),\ \max(y_m+p,\ y_n+p)] & \text{when (ii)} \end{cases}$$

$$I_5 = \ (\max(y_m+p,\ y_n+p),\ \infty) \tag{6}$$

where y_m, y_n are some constants and (i), (ii) are the conditions $|y_m-y_n| > 2p$ and $|y_m-y_n| \le 2p$ respectively. Let n_j, $j = 1, 2, \ldots, 5$, be non-negative integers with $0 \le n_j \le L$ and $\sum_{j=1}^{5} n_j = L$. Let E_j be the set of X_i's in Q with $n\text{-}N + \sum_{h=1}^{j-1} n_h \le i \le n\text{-}N + \sum_{h=1}^{j} n_h - 1$ for $j = 1, 2, \ldots, 5$. Now for the X_i's in M_m or M_n define A_m as the set of (X_i-y_m)'s with $m\text{-}N \le i \le m\text{-}N + k_m - 1$ and A_n is the set of (X_i-y_n)'s with $m+N+1 \le i \le m+N + k_n$, where $0 \le k_m,\ k_n \le 2N+1-L$. Finally let us define the following probabilities:

$$P_j^{\mu}[c,[a,b],\ (A_\mu,\ E_j)] = P[\sum_{A_\mu \cup E_j} (X_i-y_\mu) \le c,\ |X_i-y_\mu| \le p$$

$$\forall (X_i-y_\mu)\varepsilon A_\mu,\ a \le X_i-y_\mu \le b\ \forall X_i \varepsilon E_j] \tag{7}$$

when $A_\mu \cup E_j \ne \phi$, and zero otherwise, where c, a, b are some constants with $a < b$, μ is m or n and j is 2 or 4. When $a = -p$ and $b = p$, $P_j^{\mu}[c,[a,b],\ (A_\mu,\ E_j)]$ is obtained from (4) by substituting $k_\mu + n_j$ for k. We can also determine $P_j^{\mu}[c,[a,b],\ (A_\mu,\ E_j)]$ when either a or b is different from -p or p by simple modifications of the procudure to obtain P_J in Section II.

Now the output bivariate distribution function is given by

$$\begin{aligned} P(Y_m \le y_m,\ Y_n \le y_n) &= P(\sum_{M_m \cup Q} \phi(X_i - y_m) \le 0,\ \sum_{M_n \cup Q} \phi(X_i-y_n) \le 0) \\ &= \hat{\sum} \hat{C}\ P(\sum_{M_m \cup Q} \phi(X_i-y_m) \le 0,\ \sum_{M_n \cup Q} \phi(X_i-y_n) \le 0 \,| \\ &\qquad \text{all } X_i\text{'s in } E_j \text{ lie in } I_j \text{ for } j = 1,2,..,5) \\ &\qquad \cdot \prod_{j=1}^{5} P(\text{all } X_i\text{'s in } E_j \text{ lie in } I_j) \end{aligned} \tag{8}$$

where $\hat{\sum} = \sum_{n_1=0}^{L} \sum_{n_2=0}^{L_2} \sum_{n_3=0}^{L_3} \sum_{n_4=0}^{L_4}$, $\quad L_h = L - \sum_{j=1}^{h-1} n_j$

and $\hat{C} = \binom{L}{n_1}\binom{L_2}{n_2}\binom{L_3}{n_3}\binom{L_4}{n_4}$. In order to determine the bivariate distribution we need to examine four separate cases. Two of these when $y_n \ge y_m$ are considered further in the Appendix. The bivariate distributions when $y_n < y_m$ can be determined similarly.

IV. OUTPUT DISTRIBUTIONS FOR MEDIAN AND RUNNING MEAN FILTERS

It is interesting to determine the output uni- and bivariate distributions of the median and running mean filters using the general expressions obtained in the previous sections.

A. Median Filter

Univariate distribution: Consider (2). We note that as p goes to 0 P_J goes to 0 and only the term corresponding to k = 0 remains. Since $F_z(z) = F_x(z+y)$,

$$P(Y_{N+1} \le y) = \sum_{\ell=0}^{N} \binom{2N+1}{\ell} [1-F_x(y)]^{\ell} [F_x(y)]^{2N+1-\ell}$$

$$= \sum_{j=N+1}^{2N+1} \binom{2N+1}{j} [1-F_x(y)]^{2N+1-j} [F_x(y)]^{j} \tag{9}$$

This is the well-known expression for the distribution of the sample median when the input is i.i.d.

Bivariate distribution: Note that as p goes to 0, $|y_n - y_m| \ge 2p$ become always true. Thus from (A2) in the Appendix, we see that $P_j^{\mu}[c,[-p,p], (A_{\mu}, E_j)]$ approaches 0 as p approaches 0 and only the term corresponding to $k_{\mu} + n_j = 0$ remains. Now from (A1),

$$P(Y_m \le y_m, Y_n \le y_n) = \sum_{n_1=0}^{L} \sum_{n_3=0}^{L-n_1} \binom{L}{n_1}\binom{L_3}{n_3} \cdot P(\sum_{M_m} \phi(X_i - y_m) \le c_m)$$

$$\cdot P(\sum_{M_n} \phi(X_i - y_n) \le c_n) \cdot \{F_x(y_m)\}^{n_1}$$

$$\cdot \{F_x(y_n) - F_x(y_m)\}^{n_3} \cdot \{1-F_x(y_n)\}^{n_5} \tag{10}$$

where $c_m = (n_1 - n_3 - n_5)p$, $c_n = (n_1 + n_3 - n_5)p$. Here, from (A2),

$$P(\sum_{M_{\mu}} \phi(X_i - y_{\mu}) \le c_{\mu}) = \sum_{\ell_{\mu}=0}^{\hat{L}_{\mu}} \binom{2N+1-L}{\ell_{\mu}} \{F_x(y_{\mu})\}^{\omega_{\mu}}$$

$$\cdot \{1-F_x(y_{\mu})\}^{\ell_{\mu}} \tag{11}$$

where $\hat{L}_{\mu} = \min[2N+1-L, \frac{1}{2}(c_{\mu}/p + 2N+1-L)]$. Substituting (11) into (10) we obtain the output bivariate distribution of a median filter which can be seen to be the same as the one given in [7].

B. Running Mean Filter

Univariate distribution: As p approaches ∞ the term corresponds to k = 2N+1 and ℓ = 0 is the only term to be considered in (2). For this we have c = 0, and

$$P(Y_{N+1} \le y) = P(\sum_{i=1}^{2N+1} (X_i - y) \le 0).$$

This is obviously the output univariate distribution of a running mean filter.

Bivariate distribution: Now $|y_n - y_m| < 2p$ is always true as $p \to \infty$. Thus from (A3) $n_1 = n_2 = n_4 = 0$, $n_3 = L$ and $k_\mu = 2N+1-L$, $\ell_\mu = 0$ is the only term to be considered.

$$P(Y_m \le y_m,\ Y_n \le y_n) = P\Big(\sum_{i=m-N}^{m+N} (X_i - y_m) \le 0,\ \sum_{i=n-N}^{n+N} (X_i - y_n) \le 0\Big).$$

This result for the bivariate distribution of a running mean filter is obviously true by definition.

V. EXAMPLES

In this section we present the output power spectral densities for STMS's with different p values and window sizes when the input random variables are discrete valued. In particular the uniform distribution is used as the underlying distribution for the input random variables. We used discrete uniform distributions with {-1,+1} and {-1,0,+1} as input alphabets. The autocorrelations $R(\tau) = E(Y_m, Y_{m+\tau})$, for STMS's with window sizes 3, 5 and various p values are evaluated by using the bivariate distribution functions derived in the previous sections. Then the normalized power spectral densities were calculated as in [7], i.e.,

$$\Phi(w) = \sum_{\tau=-\infty}^{\infty} \frac{R(\tau)}{R(o)} e^{-j\tau w}.$$

Figures 1-4 are the plots for different window and alaphabet sizes. All of these show a low-pass characteristic for the STMS's, as expected, when the input is i.i.d. We can see that the normalized power spectral densities for STMS's which are neither the median filter nor the running mean filter are in between those of the running mean and the median filters. The running mean filter has the best low-pass characteristic, but cannot preserve edges and reject impulse noise as can the median filter. The median filter, on the other hand, may not have sufficient degree of low-pass linear action. The STMS's can be designed to give better low-pass action than the median filter and better nonlinear edge-preservation and impulse-supression characteristics than the running-mean filter. The results of this paper on the low-pass action of the STMS complement those on its nonlinear behavior [6] and are thus important for specific designs.

REFERENCES

[1] J. W. Tukey, "Exploratory Data Analysis", Reading, 1977.

[2] L. R. Rabiner, M. R. Sambur and C. E. Schmidt, "Applications of a Nonlinear Smoothing Algorithm to Speech Processing", IEEE Trans. Acoustics, Speech and Signal Proc., Vol. ASSP-23, pp. 552-557, Dec. 1975.

[3] N. S. Jayant, "Average and Median-Based Smoothing Techniques for Improving Speech Quality in the Presence of Transmission Errors", IEEE Trans. Communications, Vol. COM-24, pp. 1043-1045, Sept. 1976.

[4] W. K. Pratt, Digital Image Processing, John Wiley and Sons, 1978.

[5] A. C. Bovik, T. S. Huang, D. C. Munson, "A Generalization of Median Filtering Using Linear Combinations of Order Statistics", Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Proc., pp. 2067-2070, 1982.

[6] Y. H. Lee, S. A. Kassam, "Some Generalizations of Median Filters", Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Proc., pp. 411-414, 1983.

[7] F. Kuhlmann, G. L. Wise, "On Second Moment Properties of Median Filtered Sequences of Independent Data", IEEE Trans. Communications, Vol. COM-29, pp. 1374-1379, Sept. 1981.

APPENDIX

Case (i): Assume $y_n - y_m > 2p$. Then from (8), using (6)

$$P(Y_m \leq y_n, Y_n \leq y_n) = \hat{\sum} \hat{C} \quad P\Big(\sum_{M_m \cup E_2} \phi(X_i - y_m) \leq c_m, \; -p \leq X_i - y_m \leq p \;\; \forall X_i \varepsilon E_2\Big)$$

$$P\Big(\sum_{M_n \cup E_4} \phi(X_i - y_n) \leq c_n, \; -p \leq X_i - y_n \leq p \quad \forall X_i \varepsilon E_4\Big)$$

$$\cdot \{F_x(y_m - p)\}^{n_1} \{F_x(y_n - p) - F_x(y_m + p)\}^{n_3} \; \{1 - F_x(y_n + p)\}^{n_5} \qquad \text{(A1)}$$

where $c_m = (n_1 - n_3 - n_4 - n_5) \cdot p$ and $c_n = (n_1 + n_2 + n_3 - n_5) \cdot p$.

Now

$$P\Big(\sum_{M_\mu \cup E_j} \phi\,(X_i - y_\mu) \leq c_\mu \;\; -p \leq X_i - y_\mu \leq p \;\forall X_i \varepsilon E_j\Big)$$

$$= \sum_{\ell_\mu = 0}^{\hat{L}_\mu} \binom{2N+1-L}{\ell_\mu} \{F_x(y_\mu - p)\}^{2N+1-L-\ell_\mu} \; \{1 - F_x(y_\mu + p)\}^{\ell_\mu}$$

$$+ \sum_{k_\mu = 0}^{2N+1-L} \; \sum_{\ell_\mu = 0}^{2N+1-L-k_\mu} \binom{2N+1-L}{k_\mu} \binom{2N+1-L-k_\mu}{\ell_\mu} P_j^\mu[c, [-p, p], \; (A_\mu, E_j)] \cdot$$

$$\{F_x(y_\mu - p)\}^{w_\mu} \cdot \{1 - F_x(y_\mu + p)\}^{\ell_\mu} \qquad \text{(A2)}$$

where $\mu = m$ or n, $j = 2$ or 4, $w_\mu = 2N+1-L-k_\mu - \ell_\mu$, $\hat{L}_\mu = \min[2N+1-L, \; 1/2\, ({}^{c}\mu/p + 2N+1-L)]$ and $c = c_\mu + (w_\mu - \ell_\mu) \cdot p$. The first term in the right hand side of (A2) corresponds to the case when $A_\mu \cup E_j = \phi$. Substituting (A2) into (A1) we obtain the bivariate distribution.

Case (ii): Assume $0 < y_n - y_m < 2p$. Then

$$P(Y_m \leq y_m, \; Y_n \leq y_n) = \hat{\sum} \; \hat{c} \; \Big[\int_{y_n - p}^{y_m + p} \cdots \int_{y_n - p}^{y_m + p} P\Big(\sum_{M_m \cup E_2 \cup E_3} \phi(X_i - y_m) \leq c_m,$$

$$-p \le X_i - y_m \le y_n - y_m - p \ \forall X_i \varepsilon E_2 | X_i\text{'s in } E_3)$$

$$\cdot P(\sum_{M_n \cup E_4 \cup E_3} \phi(X_i - y_n) \le c_n,\ y_m - y_n + p \le X_i - y_n \le p \forall X_i \varepsilon E_4 | X_i\text{'s in } E_3)$$

$$\cdot \prod_{h=1}^{n_3} f(x_h) dx_1 \ldots dx_{n_3}] \cdot \{F_x(y_m - p)\}^{n_1} \ \{1 - F_x(y_n + p)\}^{n_5} \qquad \text{(A3)}$$

where $c_m = (n_1 - n_4 - n_5) \cdot p$, $c_n = (n_1 + n_2 - n_5) \cdot p$. Here $P(\sum_{M_\mu \cup E_j \cup E_3} \phi(X_i - y_\mu) \le c_\mu,$ $a \le X_i - y_\mu \le b \ \forall X_i \varepsilon E_j | X_i\text{'s in } E_j)$ can be obtained following the derivation for (A2). Note that when $y_n - y_m = 2p$ or $y_n = y_m$ the corresponding bivariate distributions can be determined from (A1) by setting $n_3 = 0$ or from (A3) by setting $n_2 = n_4 = 0$ respectively.

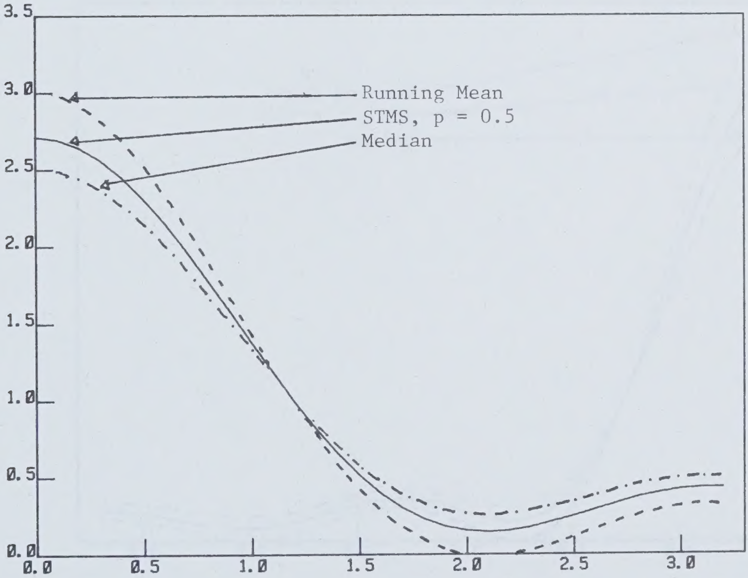


Figure 1 Normalized Power Spectra for Alphabet Size 2, 2N+I = 3.

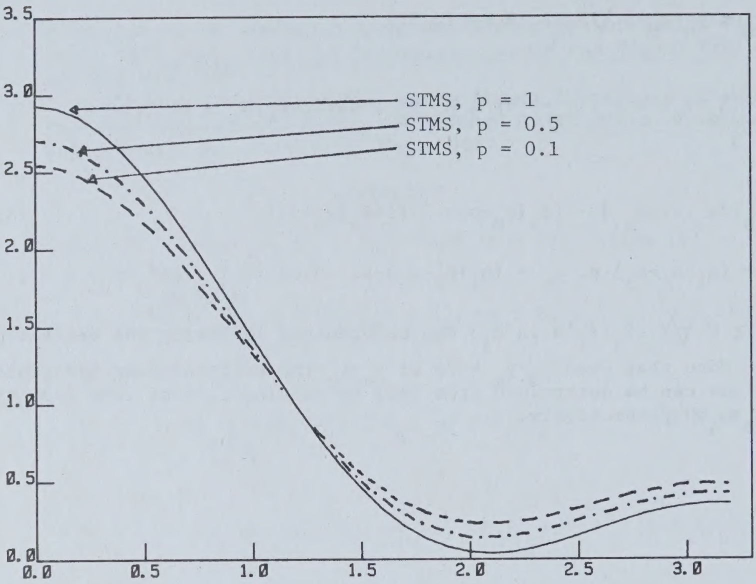


Figure 2 Normalized Power Spectra for Alphabet Size 2, 2N+1 = 3.

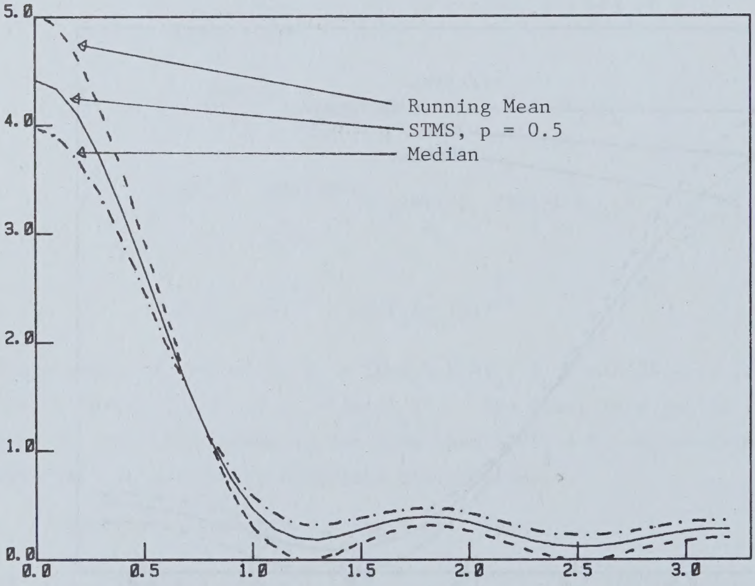


Figure 3 Normalized Power Spectra for Alphabet Size 2, 2N+1 = 5.

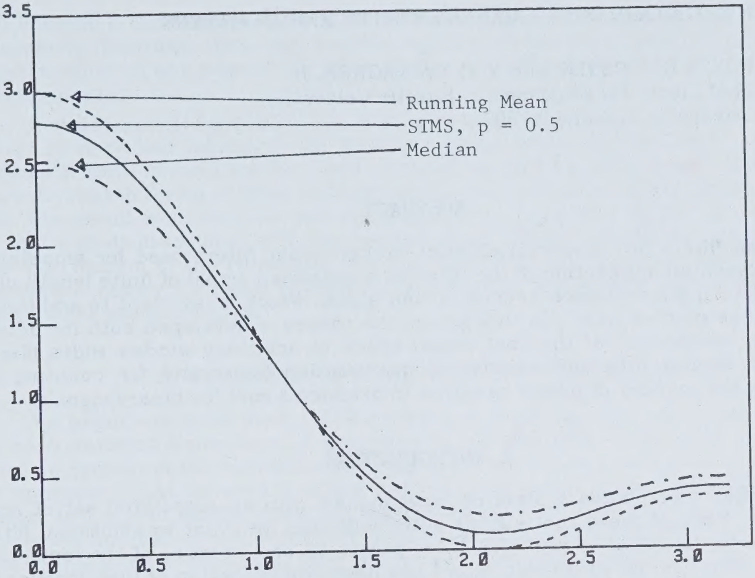


Figure 4 Normalized Power Spectra for Alphabet Size 3, 2N+1 = 3.

ROOT PROPERTIES AND CONVERGENCE RATES OF MEDIAN FILTERS

J. P. FITCH, E. J. COYLE, and N. C. GALLAGHER, Jr.
School of Electrical Engineering, Purdue University
West Layayette, Indiana 47907

ABSTRACT

Median filters are a special class of ranked order filters used for smoothing signals. Repeated application of the filter on a quantized signal of finite length ultimately results in a sequence, termed a root signal, which is invariant to additional passes of the median filter. In this paper, the theory is developed both for determining the cardinality of the root signal space of arbitrary window width filters applied to signals with any number of quantization levels and for counting or estimating the number of passes required to produce a root for binary signals.

1. INTRODUCTION

A median filter maps a class of input signals into an associated set of root sequences. Each of these root signals is by definition invariant to additional filter passes and is the result of repeated filter passes on one or more of the input signals. One effective use of median filters has been the reduction of high frequency and impulsive noise in digital images without the extensive blurring and edge destruction associated with linear filters [1]. Other applications include the smoothing of noisy pitch contours in speech signals and data compression using the root signal properties combined with a block truncation coding (BTC) technique [2,3]. In practice, processing and delay times, possible coding schemes, and output signal space characteristics can all be influenced by both the number of roots and the number of filter passes necessary to reach a root.

The implementation of a median filter requires a simple nonlinear digital operation. To begin, take a sampled signal of length L; across this signal slide a window that spans 2N+1 points. The filter output at each window position is given the same position as the sample point at the center of the window and is set equal to the median value of the 2N+1 signal samples in the window. Start up and end effects are accounted for by appending N samples to both the beginning and the end of the sequence. The front appended samples are given the value of the first signal sample; similarly, the rear appended samples receive the value of the last sample of the signal. Note that the window is moved from left to right across the signal. The basic idea is to rank the samples in the window and select the median value as the filter output.

Gallagher and Wise [4] obtained some theoretical results relating signal length to the maximum number of filter passes required to reach a root. Because our current work extends these results, several of the definitions and theorems are worth listing here.

1)A *constant neighborhood* is at least N+1 consecutive identically valued points.

2)An *edge* is a monotonic region between two constant neighborhoods of different value. The connecting monotonic region cannot contain any constant neighborhood.

3)A *root signal* is a sequence which is invariant to the median filter.

Theorem 1: Given a length L, k valued sequence to be median filtered with a 2N+1 window, a necessary and sufficient condition for the signal to be invariant under median filtering is that the appended signal consist only of constant neighborhoods and edges.

Theorem 2: Upon successive median filter window passes, any nonroot signal will become a root after a maximum of $(L-2)/2$ for L even and $(L-1)/2$ for L odd, successive filterings. Also, any nonroot signal cannot repeat, and the first point to change value on any pass of the filter window will remain constant upon successive window passes.

A state model describing the tree structure of the root signal set was proposed by Arce and Gallagher [3]. In the following section of this paper, we develop a general state description for the root signal set without using trees. Our model is unrestricted in terms of filter window widths and number of signal quantization levels. The result is a complete and exact system of equations for finding the number of root signals associated with any median filter. In the final section of this paper, the distribution of the remaining nonroot signals, in terms of the number of filter passes necessary to reach a root, is investigated for binary signals and window width three median filters.

2. STATE DESCRIPTION FOR THE ROOT-SIGNAL SET

We begin our state model with a signal of length one having k possible levels. At each iteration a new term is appended to the sequence. If this longer signal is a root we remain in the root state system, otherwise we leave to the absorbing state of nonroots. That is, once a signal contains a nonroot section, adding additional values to the end cannot make the signal a root. The value of a state at iteration L-1 is the number of root signals of length L which currently satisfy that particular state's conditions. Summing over all states at some iteration gives the number of roots associated with the filter for the specified window width, number of quantization levels, and signal length. In this manner a system of linear equations is developed to describe the number of root signals from a sequence with a fixed, but arbitrary, number of levels.

Suppose the median filter uses a window of 2N+1 points. Then by definition, a constant neighborhood is at least N+1 consecutive identically valued points. The general window size state model requires information about the final N+1 values of the signal and Theorem 1 is used to develop the state transition equations. Begin with a signal which can take on the values of the k integers from zero to k-1. In order to stay in a root region the last N+1 bits of the signal must either be monotone or constant. Our model requires two types of states: static states which are associated with constant neighborhoods and transitional states which describe edge regions. Specifically, if the last N+1 bits of the signal are monotone or constant with values $a_1, a_2, \ldots, a_{N+1}$, then our state definitions are as follows

Static States: We say the L-length signal is in static state $S_L(i+1)$, $0 \leq i \leq k-1$, if and only if $a_1 = a_{N+1} = i$.

Transitional States: An L-length signal is in an upward transitional state $U_L(j,i)$, $1 \leq j \leq N$, $1 \leq i \leq k-1$ if and only if $a_{N+1-j} < a_{N+2-j} = a_{N+1} = i$ That is, a_{N+1-j} is less than i and the j bits following a_{N+1-j} have value i. Similarly for monotone decreasing $a_1, a_2, \ldots, a_{N+1}$, we say the signal is in downward transitional state $D_L(j,i)$, $1 \leq j \leq N$, $1 \leq i \leq k-1$ if and only if $a_{N+1-j} > a_{N+2-j} = a_{N+1} = k-1-i$.

In the transitional states $U_L(j,i)$ and $D_L(j,i)$, we call j the delay parameter. For window width three filters, N=1 and j takes only one value. Note that no states are defined when $a_1, a_2, \ldots, a_{N+1}$ is neither a constant nor a monotone sequence because the signal could no longer be in the root space.

As an example of our state definitions consider the three level signal with a filter of window width seven, *i.e.*, k=3 and N=3. Suppose the last four bits of some root sequence of length L-1 are 0111. The signal is therefore currently in state $U_{L-1}(3,1)$. Note that the delay parameter specifies the length of the constant region at the end of the length N+1 sequence. Examining the results of adding different values to make this a length L sequence we see that a 2 appended to the end puts the signal in state $U_L(1,2)$ while a 1 as the new bit moves the signal into static state $S_L(2)$. Note that when a 0 is added to the signal of length L-1 it creates a nonroot region, 01110, which moves the signal into an absorbing state outside the root space.

Using their definitions, the recursive relationships among the states can be written down by inspection. We have the static states

$$S_{L+1}(i) = \begin{cases} S_L(1) + D_L(N,k-1) & ,\text{if } i=1 \\ S_L(i) + U_L(N,i-1) + D_L(N,k-i) & ,\text{if } 2 \le i \le k-1 \\ S_L(k) + U_L(N,k-1) & ,\text{if } i=k \end{cases} \tag{1}$$

and the transitional states

$$U_{L+1}(j,i) = \begin{cases} S_L(1) & ,\text{if } j=1 \text{ and } i=1 \\ \sum_{r=1}^{i} S_L(r) + \sum_{r=1}^{i-1} \sum_{q=1}^{N} U_L(q,r) & ,\text{if } j=1 \text{ and } 2 \le i \le k-1 \\ U_L(j-1,i) & ,\text{if } 2 \le j \le N \text{ and } 1 \le i \le k-1 \end{cases} \tag{2}$$

with

$$D_{L+1}(j,i) = \begin{cases} S_L(k) & ,\text{if } j=1 \text{ and } i=1 \\ \sum_{r=1}^{i} S_L(k+1-r) + \sum_{r=1}^{i-1} \sum_{q=1}^{N} D_L(q,r) & ,\text{if } j=1 \text{ and } 2 \le i \le k-1 \\ D_L(j-1,i) & ,\text{if } 2 \le j \le N \text{ and } 1 \le i \le k-1 \end{cases} \tag{3}$$

The initial conditions of the system are specified by $S_1(i) = 1$, for $i=1,\dots,k$ and $U_1(j,i) = D_1(j,i) = 0$, for all i and j. Let the number of roots to a k-level signal of length L be defined as $R^k(L)$, then we have

$$R^k(L) = \sum_{i=1}^{k} S_L(i) + \sum_{i=1}^{k-1} \sum_{j=1}^{N} (U_L(j,i) + D_L(j,i)) \tag{4}$$

This specifies the system of state equations necessary to compute the number of roots for signals of arbitrary length and number of levels. Closed form analytic solutions can be obtained by specifying the values of k and N. However these computations are not only tedious but also must be repeated whenever the number of levels or the filter window width is changed. We found that computer solution is easily accomplished. Note that symmetries between upward and downward transition states as well as tree structure symmetries among the static states can be exploited for a reduction in the amount of computation.

3. NUMBER OF FILTER PASSES TO A ROOT

The state model presented in the previous section is an exact and complete solution to counting the number of roots for arbitrary level signals and median filters of any window size. In this section of the paper we begin an investigation of the distribution of signals requiring one or more filter passes to a root. Theorem 2, in the introduction, states that the maximum number of filter passes for a fixed length signal is $(L-2)/2$ for L even and $(L-1)/2$ for L odd. By decomposing signals into root and nonroot regions, the number of filter passes necessary to reach a root is determined by the longest nonroot region. This result follows from the invariance of the root regions ,as noted by Arce and Gallagher [3], and by applying Theorem 2. Suppose the nonroot regions have lengths $k_1, k_2, \dots, k_n$ where k_m is the maximum, $1 \le m \le n$. Theorem 2 tells us that each of these sections will require either $(k_i-1)/2$ or $(k_i-2)/2$ passes to become a root region. Therefore, the largest number of passes is required by the section with length k_m. We would like to model binary signals in terms of these different root and nonroot regions. To keep the description of root and nonroot regions relatively simple to model, we restrict our attention to filters of window width three and binary sequences.

An appropriate model must generate binary signals such that the length of the longest oscillation is known. Our approach is to use the familiar "success runs" model from the theory of stochastic processes. At each iteration of a success runs model a Bernouilli experiment with probability p of success is performed. If there have been i consecutive successes the stochastic process is said to have value i. We relate the model to our signals by noting that there is a one to one correspondence between the length of the longest binary oscillation in the signal and the maximal success run of the model. Specifically, by letting p be the probability that the next bit of the signal continues an oscillation we can use the distribution of maximal runs for our computations. Here a trial of the system corresponds to the addition of a bit to the signal. If a success occurs the new bit is the complement of the current bit and if a failure occurs the new bit has the same value as the current bit. Exact expressions are not easily derived for the distribution of maximal runs. There are, however, two cases where we can arrive at some meaningful expressions. The first case, $p=1/2$, is where all sequences are equally likely. We can treat this as a deterministic problem and count the number of signals requiring k passes to a root. In the second case, we approximate the probability that no runs of length k occur and use this to generate identical results as in case one as well as distributions and expected numbers of filter passes to a root for $0<p<1$. We should note that there is a one to one correspondence between binary oscillations in the signal space and "runs" in our Markov model.

3.1 Deterministic Case

This is a combinatorial analysis of the number of signals of length L requiring k median filter passes to a root. In terms of our probabalistic model we would say that each bit assumes either value with equal probability and is independent of the other bits. Equations are derived for the special case of $k \geq L/2$. An identity, recursive on both L and k, is developed allowing a table of results to be computed. We begin with a few definitions.

1) L is the length of the signal which is arbitrary, but fixed.

2) $n_L(k)$ is the number of signals of length L with longest binary oscillation (or run) of length k.

3) $\pi_L(i)$ is the number of L-length signals requiring i passes to a root

Recall that for a binary oscillation of length k, the number of filter passes necessary to reach a root is $(k-1)/2$ for k odd and $(k-2)/2$ for k even. Therefore if k is odd, the $n_L(k+1)$ signals with longest binary oscillation of length $(k+1)$ require the same number of passes to a root as the $n_L(k)$ signals with longest oscillation of length k. This implies that

for L even:

$$\pi_L(k)=n_L(2k+2) + n_L(2k+1) \quad ,\text{if } 0 \leq k \leq (L-2)/2 \tag{5}$$

and for L odd:

$$\pi_L(k) = \begin{cases} n_L(L) & ,\text{if } k=(L-1)/2 \\ n_L(2k+2) + n_L(2k+1) & ,\text{if } 0 \leq k \leq (L-3)/2 \end{cases} \tag{6}$$

Note that $\pi_L(0)$ is the number of root signals of length L for this window width three median filter.

In order to count the number of signals of equal length requiring k filter passes to a root, we want to solve for the $n_L(k)$ as defined previously. First note that all oscillations can be divided into two classes:

1) *Butted Oscillation*- the first and/or last bit of the signal is a point in the oscillation.

2) *Nested Oscillation*- the oscillation is terminated on both sides by root regions in the signal - *i.e.*, it is not butted.

If k is greater than L/2, then these two classes partition the group of all L-length signals with maximum binary oscillation of length k into two mutually exclusive sets. Let $B_L(k)$ be defined as the number of signals with maximum binary oscillation of length k and of the butted type. Similarly, let $N_L(k)$ denote the number of signals with maximum oscillation length k that are nested. For k less than L/2, there can be more than one oscillation of length k and consequently, the signal may not be uniquely defined as nested or butted. So we define $b_L(k)$ to be the number of signals with a maximum oscillation length of k where at least one of these oscillations of k bits includes the last bit of the signal. Then for $L/2 < k < L$, $B_L(k)$ is twice $b_L(k)$. Obviously the k-length oscillation in $b_L(k)$ can terminate on either value of a bit. Also, because $L-(k+1) < k$ we know that first $L-(k+1)$ bits of the signal can take on all permutations and the total sequence will still be included in $b_L(k)$. Then, including the special case of k=L we have

$$b_L(k) = \begin{cases} 2 & \text{,if } k=L \\ 2^{L-k} & \text{,if } L/2 \le k < L \end{cases} \tag{7}$$

and

$$B_L(k) = \begin{cases} 2 & \text{,if } k=L \\ 2^{L-k+1} & \text{,if } L/2 < k < L \end{cases} \tag{8}$$

For nested oscillations greater than half the signal length, k+2 bits are fixed (oscillation must terminate within the signal), and this oscillation may assume any position among the remaining L-(k+2) bits. Requiring the oscillation to begin with a zero will decrease the number of possible signals by a factor of two. The number of ways to place our specified k+2 bits among the remaining L-(k+2) bits is L-(k+2)+1 choose 1. Because the oscillation length is greater than half the signal length, all permutations of the L-(k+2) are acceptable in keeping the maximum oscillation length at k. So for $L/2 < k < L-1$,

$$\begin{aligned} N_L(k) &= 2 \cdot \binom{L-(k+2)+1}{1} \cdot 2^{L-(k+2)} \\ &= (L-(k+2)+1) \cdot 2^{L-k-1} \\ &= (L-k-1) \cdot 2^{L-k-1} \end{aligned} \tag{9}$$

because there can be no nested oscillation of lengths L or L-1, $N_L(k)$ is zero for k equal to L or L-1. Conclusion, for $L/2 < k \le L$,

$$\begin{aligned} n_L(k) &= B_L(k) + N_L(k) \\ &= \begin{cases} 2 & \text{,if } k=L \\ 4 & \text{,if } k=L-1 \\ 2^{L-k+1} + (L-k-1) \cdot 2^{L-k-1} & \text{,if } L/2 < k < L-1 \end{cases} \end{aligned} \tag{10}$$

The final special case to be considered before we construct our recursive formula occurs when k is exactly half the signal length. Because $b_L(L/2)$ includes both the signals containing a pair of length L/2 oscillations, taking twice $b_L(L/2)$ for $B_L(L/2)$ would count these signals twice. Therefore,

$$\begin{aligned} B_L\left(\frac{L}{2}\right) &= 2 \cdot b_L\left(\frac{L}{2}\right) - 2 \\ &= 2^{\frac{L}{2}+1} - 2 \end{aligned} \tag{11}$$

and

$$N_L\left(\frac{L}{2}\right) = \left(L - \frac{L}{2} - 1\right) \cdot 2^{L-\frac{L}{2}-1} \tag{12}$$

$$= \left[\frac{L}{2} - 1\right] \cdot 2^{\frac{L}{2} - 1}$$

So for L even,

$$n_L\left[\frac{L}{2}\right] = 2^{\frac{L}{2}+1} + \left[\frac{L}{2} - 1\right] \cdot 2^{\frac{L}{2}-1} - 2 \tag{13}$$

This leaves the derivation of $n_L(k)$ for $k < L/2$. Begin by assuming the value of $n_{L-1}(k)$ is known. There are exactly two ways to append a bit to the right end of a signal of length L-1. Therefore, $n_L(k)$ is approximately equal to twice $n_{L-1}(k)$. If we can subtract off the cases where the appended bit creates an oscillation of length k+1 and add on the cases where a binary oscillation of length k is created by the appended bit, we will have $n_L(k)$ specified. Recalling that $b_{L-1}(k)$ is the number of length L-1 signals with maximum binary oscillation of length k butted to the right end of the signal, then we have by definition

$$n_L(k) = 2 \cdot n_{L-1}(k) - b_{L-1}(k) + b_{L-1}(k-1) \tag{14}$$

We previously specified $b_L(k)$ for k greater than or equal to L/2, leaving the case of k<L/2. Our approach is to begin with the number of possible arrangements assuming the last k+1 bits are fixed and then subtract off the sequences which have oscillations of length greater than k. A factor of two is necessary because there are two ways to fix the k+1 bits. This is also why $b_{L-(k+1)}(k)$ has a coefficient of ½.

$$b_L(k) = 2 \cdot \left\{ 2^{L-(k+1)} - \sum_{j=k+1}^{L-(k+1)} n_{L-(k+1)}(j) - \tfrac{1}{2} \cdot b_{L-(k+1)}(k) \right\} \tag{15}$$

$$= 2^{L-k} - 2 \cdot \sum_{j=k+1}^{L-k-1} n_{L-k-1}(k) - b_{L-k-1}(k)$$

Note that whenever k is greater than L in $n_L(k)$, we assume the value of the function is zero. Summarizing our equations:

$$b_L(k) = \begin{cases} 2 & ,\text{if } k=L \\ 2^{L-k} & ,\text{if } L/2 \leq k < L \\ 2^{L-k} - b_{L-k-1}(k) - 2 \cdot \sum\limits_{j=k+1}^{L-k-1} n_{L-k-1}(j) & ,\text{if } 2 < k < L/2 \\ 2 & ,\text{if } k=2 \end{cases} \tag{16}$$

and

$$n_L(k) = \begin{cases} 2 & ,\text{if } k=L \\ 4 & ,\text{if } k=L-1 \\ 2^{L-k+1} + (L-k-1) \cdot 2^{L-k-1} & ,\text{if } L/2 < k < L-1 \\ 2^{\frac{L}{2}+1} + (\frac{L}{2} - 1) \cdot 2^{\frac{L}{2}-1} - 2 & ,\text{if } k=L/2 \\ 2 \cdot n_{L-1}(k) - b_{L-1}(k) + b_{L-1}(k-1) & ,\text{if } 2 < k < L/2 \\ 2 & ,\text{if } k=2 \end{cases} \tag{17}$$

Using the relationships between $\pi_L(k)$ and $n_L(j)$, as given in Equations (5) and (6), we can recursively solve for the number of filter passes to a root. The results are identical to the root signal table seen in Fig. 1. This technique, though exact for p=1/2, is not as versatile as the approximation described in the next section.

3.2 Success Runs Model

Returning to the success runs model we find that an approximation to the probability of no success runs of length r in n trials, which we will call $Q_n(r)$, is developed in Feller [5, pp. 322-328]. To use this approximation we need the definition of "run" used in the derivation. We say that a sequence of n letters S and F contains as many S-runs of length r as there are non-overlapping uninterrupted blocks containing exactly r letters S each. For example, SSSSFSSSSSSF contains three S runs of length 3 and also has five S runs of length 2. Now suppose we generate our binary signals using the following procedure.

1) Randomly select a starting bit

2) Let the transition probabilities be Markov with

$$
\begin{aligned}
p &= \Pr(B_{n+1}=1 \mid B_n=0) = \Pr(B_{n+1}=0 \mid B_n=1) \qquad (18)\\
q &= \Pr(B_{n+1}=1 \mid B_n=1) = \Pr(B_{n+1}=0 \mid B_n=0)\\
&= 1 - p
\end{aligned}
$$

where B_j is the value of the signal at position j, $1 \le j \le L-1$.

3) After every bit perform a Bernouilli experiment with probability p of success. Where success means an oscillation begins or continues and failure means a constant neighborhood begins or continues. Here a run of successes is an oscillation of the bits, which demonstrates the equivalence of "run" and "oscillation". For a length L signal we need L-1 Bernouilli trials.

Because we are now dealing with random sequences, we want the probability of an L-length signal requiring less than k passes of a median filter of window width three to reach a root. Call this probability $F_L(k)$. For $p=q=1/2$, all sequences are equally probable and the same results are expected as in the combinatorial analysis of the preceding section. In order to use Feller's result we require a relationship between $F_L(k)$ and $Q_{L-1}(j)$. Assume that L is odd then the maximum number of filter passes to a root is $(L-1)/2$. Obviously $F_L((L-1)/2)$ equals the probability that no run of length L-1 occurs in the L-1 Bernouilli trials; *i.e.*, $Q_{L-1}(L-1)$. Similarly,

$$
F_L\left[\frac{L-1}{2} - j\right] = \Pr(\text{No success run of length} \ge L-1-2j) \qquad (19)
$$

$$
= \begin{cases} 1 & ,\text{if } j<0 \\ Q_{L-1}(L-1-2j) & ,\text{if } j=0,1,2,\ldots,\left[\frac{L-1}{2} - 1\right] \\ 0 & ,\text{if } j>\frac{L-1}{2} - 1 \end{cases}
$$

and for L even

$$
F_L\left[\frac{L-2}{2} - j\right] = \begin{cases} 1 & ,\text{if } j<0 \\ Q_{L-1}(L-2-2j) & ,\text{if } j=0,1,2,\ldots,\left[\frac{L-2}{2} - 1\right] \\ 0 & ,\text{if } j>\frac{L-2}{2} - 1 \end{cases} \qquad (20)
$$

We now have the probability that less than k filter passes are sufficient to reach a root in terms of the $Q_{L-1}(j)$. The $F_L(k)$ may be used to calculate any desired statistics for the number of filter passes necessary to reach a root.

The approximation for the $Q_{L-1}(j)$ is easily done by computer. We use the following approach:

1) Let $x_0 = 1$

2)Iterate with $x_{m+1} = 1 + q \cdot p^j \cdot (x_m)^{j+1}$

3)This tends to some value x

4)$Q_L(j) \approx \dfrac{1 - p \cdot x}{(j + 1 - j \cdot x) \cdot q} \cdot \dfrac{1}{x^{L+1}}$

5) The absolute error is less than $\frac{2 \cdot (j-1) \cdot p}{j \cdot q \cdot (1+p)}$

The results, as seen in Fig. 1, match those previously derived by an exact combinatorial approach. But we now have additional information for signals where each bit is correlated with the preceding bit. As expected, for p tending to zero the probability of constant neighborhoods tends to one and the expected number of filter passes to a root goes to zero. Similarly, for p near one we expect that on the average the signal will require the maximum number of passes to a root as specified by Theorem 2 of section one.

This model for the signal space is used to estimate the the number of filter passes to a root for a chosen confidence level. If i is the smallest integer such that the probability of reaching a root signal in i or fewer passes is less than or equal to α then, we say that i passes are necessary to be 100α percent confident of having a root. In Fig. 2, plots for several values of α are given for signals generated using four different correlation values of p. There is a reduction from the maximum number of passes, (512-2)/2, to 32 passes with 95 percent confidence of a root when p=0.9 is the probability of nonroot regions continuing. For the case of equally probable signals, p=0.5, we see that only six filter passes are required for 95 percent root confidence when the signal length is less than or equal to 512. This is a reduction of 249 passes from the maximum, which would have a confidence of 100 percent. These examples indicate the possibility of reducing the number of filter passes from the maximum while maintaining a high confidence of reaching a root.

CONCLUSION

The convergence of binary signals to roots is analyzed for deterministic and random sequences. For arbitrary signal levels, a direct technique for explicit solution to the number of roots of a median filter with arbitrary window width is developed. Using the symmetric tree structure of the signal space we introduce a faster technique for analytic and computer solution.

REFERENCES

[1] T.S. Huang and G.J. Yang, "Median filters and their applications to image processing," School of Electrical Engineering, Purdue University, West Lafayette, IN, TR. EE 80-1 Jan. 1980.

[2] L.R. Rabiner, M.R. Sambur, and C.E. Schmidt, "Applications of a nonlinear smoothing algorithm to speech processing," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, pp. 552-557, Dec. 1975.

[3] G.A. Arce and N.C. Gallagher, Jr., "State description for the root-signal set of median filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, pp. 894-902, Dec. 1982.

[4] N.C. Gallagher and G.L. Wise, "A theoretical analysis of the properties of median filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, pp. 1136-1141, Dec. 1981.

[5] W. Feller, *An Introduction to Probability Theory and Its Applications, Volume I.* New York: Wiley, 1968.

```
g(k)=Pr{No success run of length k in n trials}
E is the expected # of filter passes to a root


 n  g(1)   g(2)   g(3)   g(4)   g(5)   g(6)   g(7)   g(8)   g(9)  g(10)
 2  .147   .766
 3  .077   .620   .885
 4  .041   .502   .814   .942
 5  .022   .406   .748   .908   .971
 6  .011   .328   .688   .875   .954   .985
 7  .006   .266   .633   .843   .938   .977   .992
 8  .003   .215   .582   .813   .922   .969   .988   .996
 9  .002   .174   .535   .783   .906   .961   .984   .994   .998
10  .001   .141   .492   .755   .891   .953   .981   .992   .997   .999

Signal              Probability of signal requiring i passes
Length    E      i=0     1      2      3      4      5      6      7
   3    .234    .766   .234
   4    .380    .620   .380
   5    .557    .502   .440   .058
   6    .686    .406   .502   .092
   7    .812    .328   .547   .110   .015
   8    .914    .266   .578   .134   .023
   9   1.007    .215   .598   .156   .027   .004
  10   1.088    .174   .609   .178   .033   .006
  11   1.160    .141   .614   .198   .039   .007   .001
  12   1.225    .114   .614   .218   .045   .008   .001
  13   1.283    .092   .609   .236   .051   .010   .002   .000
  14   1.336    .074   .601   .254   .057   .011   .002   .000
  15   1.385    .060   .591   .271   .062   .013   .002   .000   .000
  16   1.431    .049   .579   .287   .068   .014   .003   .001   .000


Signal                 Number of signals requiring i passes
Length    E       i=0       1       2       3       4       5       6
   3    .234        6       2
   4    .380       10       6
   5    .557       16      14       2
   6    .686       26      32       6
   7    .812       42      70      14       2
   8    .914       68     148      34       6
   9   1.007      110     306      80      14       2
  10   1.088      178     624     182      34       6
  11   1.160      288    1258     406      80      14       2
  12   1.225      466    2514     892     184      34       6
  13   1.283      754    4990    1936     416      80      14       2
  14   1.336     1220    9852    4162     926     184      34       6
```

Fig. 1 Binary signal results using Feller's approximation with p=0.5.

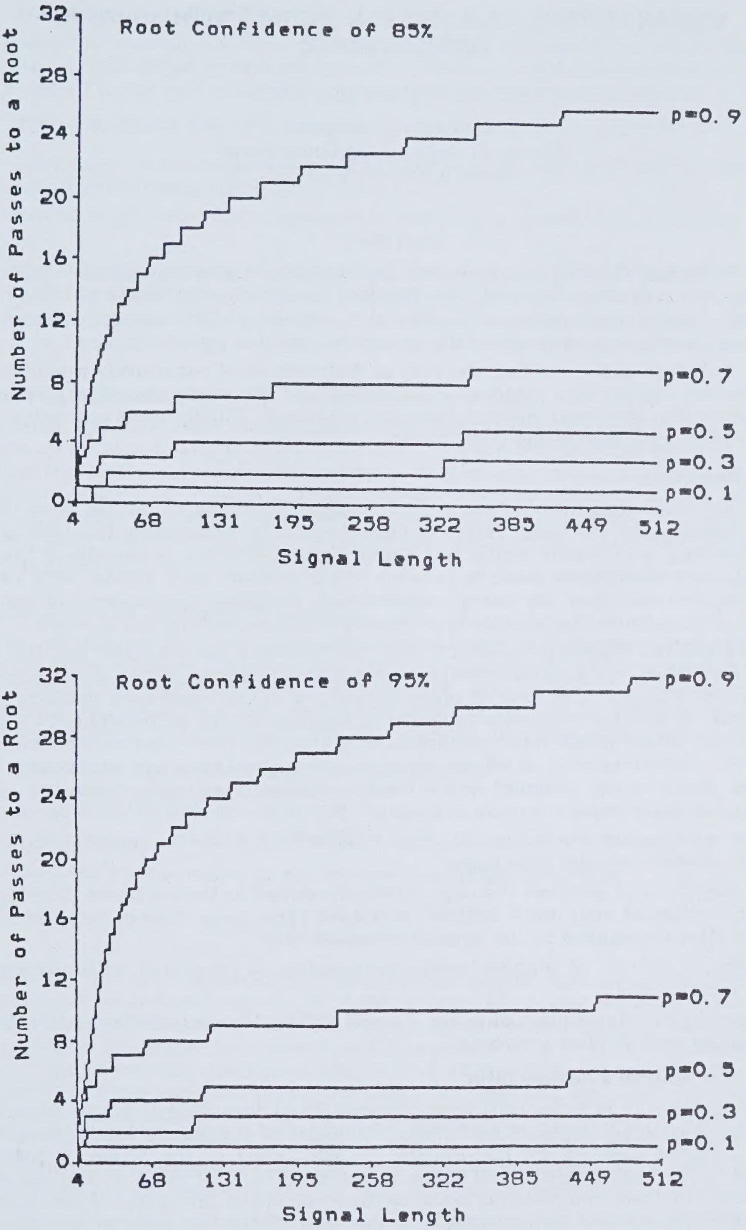


Fig. 2 Root confidence plots for binary signals generated using Feller's approximation.

MEDIAN FILTERS: ANALYSIS FOR SIGNALS WITH ADDITIVE * IMPULSE NOISE.

R.J. CRINON and G.R. ARCE

University of Delaware
Electrical Engineering Department
Newark, Delaware 19711

ABSTRACT

Median filtering is a nonlinear technique for smoothing signals. This method is finding increasing use in many areas including image processing – where edge definition is critical – and speech processing – where impulsive type noise is generally present in sampled signals.

In this paper we find the output distribution of recursively median filtered signals with additive impulsive noise. We study the edge jitter effect that recursive median filtering introduces. Finally some examples are included illustrating these results.

1. INTRODUCTION

In many data acquisition systems, such as digital imaging, impulsive noise is added to the original signal and linear filters are not suitable for eliminating this type of noise. Median filtering, a nonlinear digital technique, performs better in smoothing the signal when impulsive distributed noise is present. Median filters have proven very useful in several applications: they are easily implemented on digital processors and can do an excellent job in eliminating impulse type noise and without deteriorating edges.

There are two classes of median filters; the *Standard median filter* and the *Recursive median filter*. To date, no comparison has been carried out to determine which class is more efficient in smoothing noisy signals and at the same time tracking the original signal. To this end, two criteria can be distinguished; one is the ability to eliminate the noise; the other, is how much distortion does the filter introduce to important signal structures; structures such as edges, which are important in image processing. In this paper, we compare the standard and recursive median filter under these two criteria, when additive white impulsive noise is present. Our study consists of three parts:

- First, we consider the statistical properties of both filters on constant signals plus additive white impulse type noise.
- The second part analyzes the edge jitter introduced by the recursive filtering on an edge corrupted with white additive impulsive type noise. This is compared to the edge jitter introduced by the standard median filter.
- Finally, a criteria of window length optimization is presented to illustrate these theoretical results.

Before starting the description of some of these different parts, a review of the standard and recursive median filter is in order.

1) The standard median filter.

Median filtering is a discrete time process. We slide a 2N + 1 points wide window across an entire input signal. At each step, the output of the filter is the median value of the 2N+1 points spanned by the window. At both ends of the signal, N points are appended to the beginning and end of the signal to account for start up and end effects. The value of the front end points is equal to the value of the first point of the signal, and the value of the last end points is equal to the value of the last point of the signal. If { x(.) } and { y(.) } are the input and the output sequences respectively, then the output is:

$$y(a) = \text{median}\,[\,x(a-N), \,.....,\, x(a), \,.....,\, x(a+N)\,]\ . \tag{1}$$

---

* The authors gratefully acknowledge the support of the National Science Foundation under Grant # ECS - 830 7764

2) **The recursive median filter**

The recursive median filter is a modification of the standard filter where the value of the center sample of the window is replaced by the computed median value before the window is shifted to the next position. In other words, the filtered sequence is

$$y(a) = \text{median}\,[\ y(a-N),\ \ldots..,\ y(a-1),\ x(a),\ x(a+1),\ \ldots..,\ x(a+N)\ ]\ . \qquad (2)$$

The advantages of the recursive median filter over the standard will be mentioned after we define the following signal structures.

- A *Constant neighborhood* is a region of at least N + 1 consecutive identically valued points.
- An *Edge* is a monotonically rising or falling set of points, whose first and last points are part of two constant neighborhoods (each of different value).
- An *Impulse* is a set of no more than N consecutive points whose value is different from the two surrounding regions which are two constant neighborhood identically valued.
- A *Root* is an invariant signal to further median filtering.

Gallagher and Wise[1] have shown than at most (L-2)/2 passes of the median filter are required to reduce an L length signal to its root. The advantage of the recursive operation is that the root signal is obtained after the first pass of the recursive filter[2]; hence reducing processing time. This is desirable if real time processing is needed. Although the roots generated by the standard and recursive filter may differ for the same input signal, they have the same characteristics. In fact, roots of the recursive filter are also roots of the standard filter.

2. THE FILTERING OF IMPULSE TYPE NOISE

In this section, we first derive the output distribution of recursively median filtered signals with white, additive impulsive noise (for any window size). These results are then used to compare the efficiency of the standard filter against that of the recursive filter in eliminating the impulsive noise. Two kinds of white impulsive noise are considered; positive impulsive and two sided (positive and negative) impulsive white noise.

1) **The filtering of positive impulsive type noise.**

Consider a constant signal plus noise:

$$s(i) = c + n(i)\ , \qquad (3)$$

where in this case c is the signal (zero) and the noise is distributed as

$$P_r(n) = \begin{cases} \alpha = 1-\beta & \text{for } n = 0 \\ \beta & \text{for } n = 1\ . \end{cases} \qquad (4)$$

In order to find the statistical description of the filtered signal, we must first define some structures for the root signals. Root signals are only composed of constant neighborhoods and edges [3]. This root structure was used by Arce and Gallagher [4] to define signal states, where the states are some specific sequence of (n + 1) root sample values. For binary signals we can distinguish two sets of states. One set contains the states formed with *Constant Neighborhoods*, the other set contains the states formed with sequences containing *Edges*. Because of the binary nature of the signal in our model, the first set has only two states; call them "A" and "D". These states are described below:

State "A" consists of the sequence 000...00 (n+1 root values equal to 0)
State "D" consists of the sequence 111...11 (n+1 root values equal to 1)

In a sequence of (n+1) samples there can only be one edge, if this structure is to belong to a root. There are two kinds of edges (rising and falling), we denote the states formed with a rising edge by B, and the states formed with a falling edge by C. In a sequence of (n+1) points, there are n positions where a rising/falling edge can be located. This means that there exists n "B" states and n "C" states. B(j) denotes the state where j is the number of root values equal to "1". For example, the sequence (0 0 0 1 1) defines a "B(2)" state for a median filter of size 9. Similarly, we denote by C(j) the state where j is the number of root values equal to "0". So we have:

n states "B(j)" consisting of the sequences 00...01...11 (# of 1 = j), $1 \le j \le n$, and
n states "C(j)" consisting of the sequences 11...10...00 (# of 0 = j), $1 \le j \le n$.

Let $\{r_n(i)\}$ be a root sequence for a filter of window $2n + 1$. Define S^n_k as the (n+1)th long root sequence:

$$S^n_k = \Big[r_n(k-n), \;\cdots\;, r_n(k) \Big] . \tag{5}$$

If S^n_k is the sequence defining a state B(j-1) ($j \neq 1$), then in order for $\{ r_n(i) \}$ to be a root sequence, $r_n(k+1)$ must have a value of "1". Otherwise an impulse would occur. This implies S^n_{k+1} is the sequence defining the state B(j). If S^n_k defines B(n), then S^n_{k+1} can only define a D state. Equivalent relations can be obtained for sequences defining C(j) states. Moreover, if S^n_k defines an A state, then S^n_{k+1} can be an A state again or a B(1) state. These relations can be rearranged so that the notion of "delay" states is introduced and they are summarized below, where the subscript k in each letter state means that they are defined by the segment S^n_k.

$$\begin{array}{ll}
A_{k+1} = A_k + D_{k-n} & C(n)_{k+1} = D_{k-(n-1)} \\
B(1)_{k+1} = A_k & \quad\vdots \\
B(2)_{k+1} = A_{k-1} & C(j)_{k+1} = D_{k-(j-1)} \\
\quad\vdots & \quad\vdots \\
B(j)_{k+1} = A_{k-(j-1)} & C(2)_{k+1} = D_{k-1} \\
\quad\vdots & C(1)_{k+1} = D_k \\
B(n)_{k+1} = A_{k-(n-1)} & D_{k+1} = A_{k-n} + D_k \; .
\end{array} \tag{6}$$

All states at step (k + 1) are written as function of states consisting of constant neighborhoods. In order to reach A_{k+1}, either we have A_k as the previous state, or D_{k-n} as the state n steps ago. Likewise, D_{k+1} can be reached either from D_k or A_{k-n}. The delay n represents simply the n necessary transitions through the B or C states.

The output distribution for a recursively median filtered sequence of i.i.d. signals is now derived[5]. Consider the binary valued input source with a probability distribution as follows:

$$P_r(0) = \alpha = 1 - \beta \qquad P_r(1) = \beta \;;\; \alpha > 0\,,\; \beta > 0\;. \tag{7}$$

Each state transition probability for the root signal is the sum of the occurrence probabilities for all the sequences that once filtered, yield the desired transition. For instance, the filtered sequence:

$$\overbrace{\overbrace{\underbrace{0\,0\,\ldots\ldots\ldots\ldots\,0\,0}_{(n+1)\ samples}}^{A_k}\,1}^{B(1)_{k+1}}\;, \tag{8}$$

yields a transition from a state A_k to a $B(1)_{k+1}$ state. The conditional probability of such filtered sequence occurring is:

$$P_r\Big[B(1)_{k+1} \,/\, A_k \Big] = \frac{P_r\,(\,B(1)_{k+1}\,,\,A_k\,)}{P_r\,(\,A_k\,)} = \beta^{n+1}\;, \tag{9}$$

since the next n samples after the "1" sample in (8) must be "1"; otherwise this last sample of value one would have been filtered to a "0". Similarly, the only filtered sequence yielding the transition from D_k to $C(1)_{k+1}$ is

$$\overbrace{\overbrace{\underbrace{1\,1\,\ldots\ldots\ldots\ldots\,1\,1}_{(n+1)\ samples}}^{D_k}\,0}^{C(1)_{k+1}}\;. \tag{10}$$

occurring with the conditional probability

$$P_r\left[\,C(1)_{k+1} \,/\, D_k\,\right] = \frac{P_r\,(\,C(1)_{k+1}\,,\,D_k\,)}{P_r\,(\,D_k\,)} = \alpha^{n\,+1}\,. \tag{11}$$

From (9) and (11), and noticing that an "A" state cannot generate a "C" or "D" states, and that a "D" state cannot generate an "A" or "B" states, we obtain directly the expressions of $P_r[\,A_{k+1} \,/\, A_k\,]$ and $P_r[\,D_{k+1} \,/\, D_k\,]$. All the other non allowed transitions have a transition probability equal to zero. The transition probabilities among all the states form the stochastic matrix shown below.

$$\begin{array}{c|cccccccccc|}
 & A_{k+1} & B(1)_{k+1} & B(2)_{k+1} & B(j)_{k+1} & B(n)_{k+1} & C(n)_{k+1} & C(j)_{k+1} & C(2)_{k+1} & C(1)_{k+1} & D_{k+1} \\
A_k & 1-\beta^{n+1} & \beta^{n+1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
B(1)_k & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & & & & \vdots & \vdots & & & & & \vdots \\
B(j-1)_k & 0 & 0 & 0\ \cdots & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & & & & & \vdots & & & & & \vdots \\
B(n-1)_k & 0 & 0 & 0\ \cdots & 0\ \cdots & 1 & 0 & 0 & 0 & 0 & 0 \\
B(n)_k & 0 & 0 & 0\ \cdots & 0\ \cdots & 0 & 0\ \cdots & 0\ \cdots & 0 & 0 & 1 \\
C(n)_k & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
C(n-1)_k & 0 & 0 & 0\ \cdots & 0\ \cdots & 0 & 1 & 0 & 0 & 0 & 0 \\
 & \vdots & & & & & \vdots & & & & \\
C(j-1)_k & 0 & 0 & 0\ \cdots & 0\ \cdots & 0 & 0\ \cdots & 1 & 0 & 0 & 0 \\
 & \vdots & & & & & \vdots & \vdots & & & \\
C(1)_k & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
D_k & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \alpha^{n+1} & 1-\alpha^{n+1} \\
 & A_{k+1} & B(1)_{k+1} & B(2)_{k+1} & B(j)_{k+1} & B(n)_{k+1} & C(n)_{k+1} & C(j)_{k+1} & C(2)_{k+1} & C(1)_{k+1} & D_{k+1}
\end{array} \tag{12}$$

The stationary probabilities of the states[6] are

$$\begin{aligned}\Pi &= \lim_{k\to\infty}[P_r(A_k),P_r(B(1)_k),\ldots P_r(B(n)_k),P_r(C(n)_k),\ldots P_r(C(1)_k),P_r(D_k)].\\ &= [\,\pi(A),\,\pi(\,B(1)\,),\,\ldots,\,\pi(\,B(n)\,),\,\pi(\,C(n)\,),\,\ldots,\,\pi(\,C(1)\,),\,\pi(D)\,]\ .\end{aligned} \tag{13}$$

Since all the C(j)'s states as well as the "A" states end with a "0" sample, the output stationary distribution for a "0" and "1" valued sample are, respectively:

$$\lim_{k\to\infty} P_r(y_k=0) = \pi^*(0) = \pi(A) + \sum_{j=1}^{n}\pi(\,C(j)\,) = \frac{\alpha^{n+1} + n\beta^{n+1}\alpha^{n+1}}{\alpha^{n+1} + 2n\beta^{n+1}\alpha^{n+1} + \beta^{n+1}}\ , \tag{14a}$$

$$\lim_{k\to\infty} P_r(y_k=1) = \pi^*(1) = \pi(D) + \sum_{j=1}^{n}\pi(\,B(j)\,) = \frac{\beta^{n+1} + n\beta^{n+1}\alpha^{n+1}}{\alpha^{n+1} + 2n\beta^{n+1}\alpha^{n+1} + \beta^{n+1}}\ , \tag{14b}$$

where y_k denotes the kth sample from the filtered signal and where the stationary probabilities of the states are found with known finite state Markov chain methods[6]. Hence, for a window of length $2n + 1$, the probability of an impulse occurring at the output of the recursive filtered signal, as a function of β is given by (14b).

The previous analysis is now used to compare the recursive and standard median filters. The results for the standard median filter, which have already been computed [7], and the previous results are plotted in Fig 1, where the probability of an impulse occurring at output of the filtered noisy signal is shown. For example, the probability of an impulse occurring at any signal point is decreased from $5.0\ 10^{-2}$ to 10^{-7} by filtering the signal with a standard median filter of window 13. The same result can be obtained by filtering the noisy signal with a recursive median filter of window length 11. Figure 1 shows that the recursive filter is more efficient in smoothing the noisy signal.

2) **The filtering of positive and negative impulsive type noise.**

Consider the noisy signal:

$$s(i) = c + n(i) \ , \tag{15}$$

where in this case c = 1 is the constant signal. The noise is distributed as

$$P_r(n) = \begin{cases} \alpha = 1 - 2\beta & \text{for } n = 0 \\ \beta & \text{for } n = 1 \\ \beta & \text{for } n = -1 \ . \end{cases} \tag{16}$$

As it can be noticed from (16), one more level has been added to account for the negative impulses; the existing levels are "0", "1" and "2". For a window length 3 up to 9, the Markov matrices representing the state transition probabilities have been defined, and the stationary probabilities of each level have been computed; particularly, the probability of positive or negative impulses at the recursive median filtered output are given by $\pi(0) + \pi(2)$. The results are plotted Fig. 2. Again the performance of the recursive filter is better than the standard filter. Moreover it can be noticed that , given β and n, the probability of an impulse at the output of the recursive median filter is lower in Fig. 2 than in Fig. 1. This is due to the existence of symmetric impulsive levels; an original level in the signal can be also retrieved from a window containing the same number of positive and negative impulses. For example the probability of an impulse (either positive or negative) occurring at any signal point is decreased from 4.0 10^{-3} to 10^{-10} using a standard median filter of window length equal to 9. The same results are obtained with a recursive filter of a window length equal to 7.

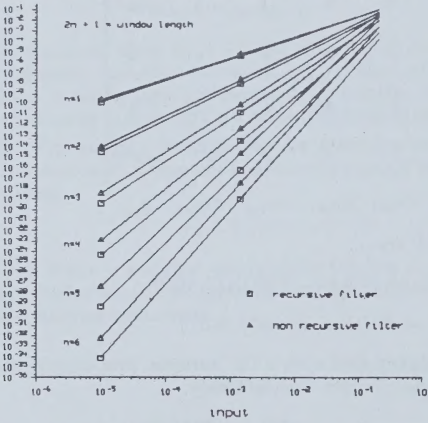


Fig 1: Probability of impulse

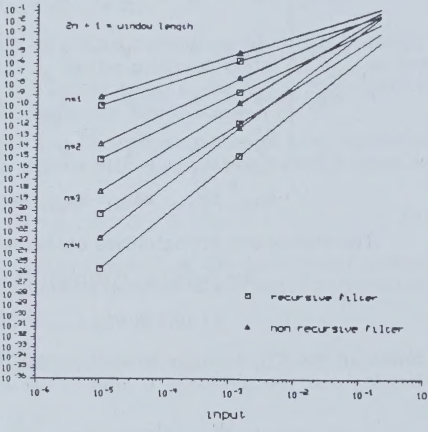


Fig 2: Probability of impulse

3. EDGE JITTER DUE TO IMPULSIVE TYPE NOISE

In this section we analyze the edge jitter introduced by the filtering of a noisy signal. As in the previous section, we begin with the analysis for white positive impulsive type noise. This analysis is then extended for positive and negative impulses.

1) **Edge jitter due to positive impulsive type noise.**

Consider a rising edge to which impulsive noise has been added (Fig 3). Let the rising edge be at position j. In other words, the noisy signal is:

$$s(i) = x(i) + n(i) \ , \tag{17}$$

where

$$x(i) = \begin{cases} 0 & \text{for } i \leq j \\ 2 & \text{for } i > j \ . \end{cases} \tag{18}$$

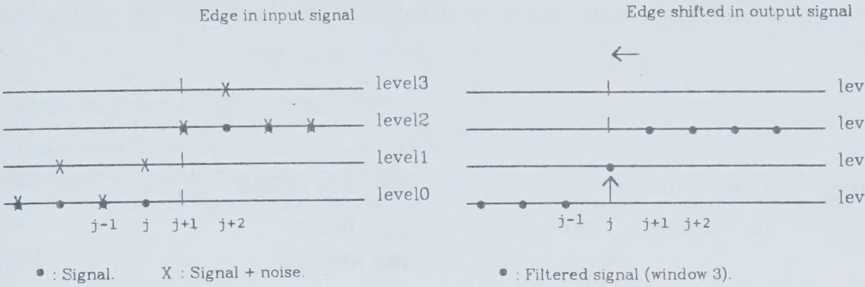


Fig 3: Impulsive noise creating edge jitter

Let the filtered sequence be y(i) and the noise be distributed as in (4). If no edge jitter occurs, the output y(j) must be equal to 0. If an impulse occurs at position j , then y(j) = 1 is part of a ***rising edge*** meaning that N + 1 points are of value ≥ y(j) and at least N + 1 points are of value ≤ y(j). Hence, the edge is shifted by one position. Figure 3 illustrates the previous example. Because of the edge on the signal, we have a nonstationary input to the filter. In fact, we have a binary stationary input signal before the window reaches the edge and after the window leaves the edge. Moreover, there are 2n points that are filtered which are nonstationary. Refer to Fig. 3. In the stationary region on the left of the edge ,the two allowed levels are "0" and "1"; the transition matrix describing the filtered output is simply (12). Again this matrix can be used up to the position (j-n), and from the position (j+n+1); however, after the edge the levels are "2" and "3" instead of "0" and "1". The 2n nonstationary input points from (j-(n-1)) to (j+n) require a point to point analysis. Each transition probability can be expressed as the sum of the occurrence probabilities for all the sequences that once filtered yield the desired transition. Let us start in the nonstationary region on the left of the rising edge. Suppose that the recursive median filter is currently at the nonstationary position (j-k), with 0 ≤ k ≤ (n-1), and consider first the transition from a signal sample y(j-k-1) = 1. To form a root signal, the next recursive filtered value at position (j-k) must be a "1", since the window contains a rising edge. Let any filtered sample be denoted by its value with its position as a subscript. Hence,

$$P_r\ (\, y(j-k)=1\ /\ y(j-k-1)=1\ ) = P_r\ (\ 1_{j-k}\ /\ 1_{j-k-1}\ ) = 1\ . \tag{19}$$

Let us consider now the transition probabilities from y(j-k-1) = 0. The sequences yielding the transition from y(j-k-1) = 0 to y(j-k) = 1 can be found using the same reasoning as in (9), with the restriction that from position (j+1) the samples have value "2" or "3". Consequently, a number of only (k+1) positions after the (j-k)th position must be of value "1" in order to obtain the desired transition; hence,

$$P_r \left[1_{j-k}\ /\ 0_{j-k-1} \right] = \frac{P_r\ (\ 1_{j-k}\ ,\ 0_{j-k-1}\ )}{P_r\ (\ 0_{j-k-1}\ )} = \beta^{k+1}\ . \tag{20}$$

There are n positions where the window contains the rising edge. Consequently, a total of n matrices are needed to express the transition probabilities from position (j-n+1) to j. Their generalized form is shown below:

$$\begin{array}{c|cc|} & 0_{j-k} & 1_{j-k} \\ 0_{j-k-1} & 1-\beta^{k+1} & \beta^{k+1} \\ 1_{j-k-1} & 0 & 1 \end{array} \quad ,\ 0 \leq k \leq (n\text{-}1) \tag{21}$$

Consider the filtered signal at position (j+1) (see Fig. 3). Since the left of the window contains n filtered samples of values "0" or "1" , (n+1) samples of value "3" are needed on the right of window to obtain a filtered "3" valued sample at position (j+1). Hence,

$$P_r \left[3_{j+1}\ /\ 0_j \right] = P_r \left[3_{j+1}\ /\ 1_j \right] = \beta^{n+1}\ . \tag{22}$$

and the transition matrix at the (j+1) stage is then:

$$\begin{array}{c|cc|} & 2_{j+1} & 3_{j+1} \\ \hline 0_j & 1-\beta^{n+1} & \beta^{n+1} \\ 1_j & 1-\beta^{n+1} & \beta^{n+1} \end{array} \tag{23}$$

Consider now the (n-1) positions (n ≠ 1) from (j+k+1) to (j+n) (for k = 1, ..., (n-1)); the window of the recursive median filter still contains the rising edge on its left. Consequently, the next root value from 3_{j+k} must be 3_{j+k+1}. Hence,

$$P_r\ (\, 3_{j+k+1} \,/\, 3_{j+k}\,) = 1 \ ,\ 1 \le k \le (n-1)\ . \tag{24}$$

Moreover, the unique sequence yielding a transition from 2_{j+k} to 3_{j+k+1} is the sequence where (n+1) "3" valued samples occur at the (n+1) rightmost positions of the window. Therefore,

$$P_r\left[3_{j+k+1} \,/\, 2_{j+k}\right] = \beta^{n+1}\ ,\ 1 \le k \le (n-1)\ . \tag{25}$$

There are (n-1) transition matrices describing the nonstationary process on the right of the rising edge. Using (24) and (25), these matrices are characterized as following:

$$\begin{array}{c|cc|} & 2_{j+1+k} & 3_{j+1+k} \\ \hline 2_{j+k} & 1-\beta^{n+1} & \beta^{n+1} \\ 3_{j+k} & 0 & 1 \end{array}\quad ,\ 1 \le k \le (n-1),\ n \ne 1 \tag{26}$$

Using the matrices (12), (21), (23), (26), we compute the output distributions for the filtered signal for all points. The edge jitter statistics are now computed. Let the value of the edge jitter be i; the variance of the edge jitter is:

$$\mathrm{Var}(i) = E\,(\,i^2\,) - \left\{E\,(\,i\,)\right\}^2\ , \tag{27}$$

where

$$E\,(\,i^k\,) = \sum_{l=0}^{l=\infty} l^k\ P_r(\,i = l\,)\ , \tag{28}$$

Consider a window length (2n+1) and an edge jitter of l for 1 ≤ l ≤ n. The probability of obtaining an edge jitter i=l is the probability that the jth to the (j-l+1)th filtered sample have value "1", and that the (n-l) remaining positions are zero valued filtered samples. The occurrence probability of this sequence is:

$$P_r(i=l) = \pi(0)\ P_r\left[0_{j-n+1} \,/\, 0_{j-n}\right] \cdots P_r\left[1_{j-l+1} \,/\, 0_{j-l}\right] \cdots P_r\left[1_j \,/\, 1_{j-1}\right] \tag{29a}$$

where π(0) represents the stationary probability of a "0" sample in the (j-n)th position. Using the matrices (12) and (21) and equation (14a), (29a) reduces to:

$$P_r(\,i = l\,) = \frac{\alpha^{n+1} + n\,\alpha^{n+1}\beta^{n+1}}{\alpha^{n+1} + 2n\,\alpha^{n+1}\beta^{n+1} + \beta^{n+1}} \left\{\prod_{k=1}^{k=n-l} (\,1 - \beta^{k+1}\,)\right\} \beta^l\ ,\ 1 \le l \le n\ . \tag{29b}$$

Likewise, the edge jitter will be equal to (n+1) when positions j to (j-n) are one valued filtered samples and positions j-(2n+1) to j-(n+1) are zero valued samples; hence, the probability of obtaining a rising edge jitter i=(n+1) is:

$$\begin{aligned} P_r(i=n+1) = P_r(0_{j-(2n+1)},\dots,0_{j-(n+1)})\ & P_r\left[0_{j-2n},\dots,0_{j-(n+1)},1_{j-n} \,/\, 0_{j-(2n+1)},\dots,0_{j-(n+1)}\right] \\ & P_r\left[1_{j-n-1} \,/\, 1_{j-n}\right] \cdots P_r\left[1_j \,/\, 1_{j-1}\right]\ . \end{aligned} \tag{30a}$$

In (30a), positions j-(2n+1) to j-(n+1) are on the stationary region to the left of the edge; therefore, (n+1) sequential samples form a state of the stationary region. Clearly, ($0_{j-(2n+1)}, \dots, 0_{j-(n+1)}$) forms a state " A_{j-2} " and ($0_{j-2n}, \dots, 0_{j-(n+1)}, 1_{j-n}$) forms a state " $B(1)_{j-1}$ ". Hence,

$$P_r(i=n+1) = \pi(A)\ \ P_r\left[B(1)_{j-n} \,/\, A_{j-(n+1)}\right] P_r\left[1_{j-n-1} \,/\, 1_{j-n}\right] \cdots P_r\left[1_j \,/\, 1_{j-1}\right]\ . \tag{30b}$$

Using (21), (12), (14a) and (14b) we find

$$P_r(\,i\, = n+1\,) = \beta^{n+1}\,\frac{\alpha^{n+1}}{\alpha^{n+1} + 2n\,\alpha^{n+1}\beta^{n+1} + \beta^{n+1}}\,. \qquad (30c)$$

Recall the notion of "delay state". Before meeting a "D" state formed of (n+1) one valued root samples, we must previously go through n states "B(j)". Thus for the edge jitter values from (n+2) up to (2n+1), we come from a state "B(1)" and from state "B(n)" respectively. Moreover, it can be shown in solving (14a) and (14b) that $\pi[B(1)] = \pi[B(2)] = \ldots = \pi[B(n)]$. Using (12), the probability that the edge jitter i = l , for $n+2 \leq l \leq 2n+1$ is:

$$P_r(\,i\, = l\,) = \frac{\alpha^{n+1}\beta^{n+1}}{\alpha^{n+1} + 2n\,\alpha^{n+1}\beta^{n+1} + \beta^{n+1}}\,. \qquad (31)$$

Finally, for $l \geq 2n+2$, the propagation of the positive impulses is only done through the state "D" and is terminated by a transition from "B(n)" to "D", the number of transition along the state "D" being (l-(2n+1)). Thus the probability of the edge jitter equal to l, with $l \geq 2n+2$ is obtained using (31), (12):

$$P_r(\,i\, = l\,) = \frac{\alpha^{n+1}\beta^{n+1}}{\alpha^{n+1} + 2n\,\alpha^{n+1}\beta^{n+1} + \beta^{n+1}}\left\{1 - \alpha^{n+1}\right\}^{(l-(2n+1))}\,. \qquad (32)$$

The case of the falling edge is also treated in a similar fashion, where position j is now the first zero or one valued position after the falling edge. There are n matrices of the form (33a) describing the nonstationary transitions on the left of the falling edge; the matrix related to position j is shown in (33b); (33c) is the expression of n matrices describing the process on the right of the falling edge.

$$\begin{array}{c|cc|} & 2_{j-k} & 3_{j-k} \\ \hline 2_{j-k-1} & 1 & 0 \\ 3_{j-k-1} & (1-\beta)^k & 1-(1-\beta)^k \end{array} \qquad \begin{array}{c|cc|} & 0_j & 1_j \\ \hline 2_{j-1} & (1-\beta)^{n+1} & 1-(1-\beta)^{n+1} \\ 3_{j-1} & (1-\beta)^{n+1} & 1-(1-\beta)^{n+1} \end{array} \qquad \begin{array}{c|cc|} & 0_{j+k} & 1_{j+k} \\ \hline 0_{j+k-1} & 1 & 0 \\ 1_{j+k-1} & (1-\beta)^{n+1} & 1-(1-\beta)^{n+1} \end{array}$$

a) with $1 \leq k \leq n$. b) c) with $1 \leq k \leq n$. (33)

To compare the standard median filtering with the recursive median filtering of a signal plus white impulsive type noise, we use the same four level model as chosen previously. After computing the statistics for the edge jitter for the standard median filter, the variance of the edge jitter is obtained for different window lengths. Table 1 shows the variance of the edge jitter for windows of size 3 and 5. Unlike the recursive filter, the variance of the rising edge and falling edge are the same when the standard median filter is used.

These results lead to the following observations: the recursive median filter preserves rising edges better than the standard median filter. Table 1 shows that, for the selected values of β, the variance of the rising edge jitter for the recursive filter is smaller than for the standard filter for both window sizes. Conversely, the variance of the falling edge jitter for the recursive median filter is slightly greater than for the standard median filter. Also the original symmetry of a rising and falling edges in an signal is not preserved by the recursive median filtering process. In order to illustrate better the nonsymmetry, the probability of each level at the output of the recursive median filter in the nonstationary region of both edges is computed. Fig 4a shows the results for the filter of window size 3.

2) **Edge jitter due to positive and negative impulsive type noise.**

Consider an original two level signal that is perturbed by either positive or negative impulses. Hence we have a six level signal with two kinds of edge jitter: a rising edge can be shifted to the left because of the occurrence of positive impulses on its left. But a rising edge can be also shifted to the right because of negative impulses occurring on its right. Again the transition matrices describing the nonstationary region are computed. However the matrices are 3x3 matrices. The variance of the rising and the falling edge are then computed for a window length 3 and 5 using a similar analysis as presented in section 1. Table 2 shows the results. Notice that the standard median filtering of a signal with double sided white impulsive type noise creates more edge jitter than the recursive median filter, regardless of the nature of the edge (rising or falling edge). This confirms that the recursive median filter performs better in preserving edges at their original position in a noisy signal than the standard filter. The statistics of each level at any nonstationary position are also computed. Figure 4b shows again a nonsymmetry between the edge jitter in a rising edge and in a falling edge (window size 3).

β	n = 1			n = 2		
	Std	Rec R	Rec F	Std	Rec R	Rec F
1.0e-05	2.0e-05	1.0e-05	2.0e-05	3.0e-05	1.0e-05	3.0e-05
1.0e-04	2.0e-04	1.0e-04	2.0e-04	3.0e-04	1.0e-04	3.0e-04
1.0e-03	2.0e-03	1.0e-03	2.0e-03	3.0e-03	1.0e-03	3.0e-03
2.0e-03	4.0e-03	2.0e-03	4.0e-03	6.0e-03	2.0e-03	6.0e-03
5.0e-03	1.0e-02	5.3e-03	1.0e-02	1.5e-02	5.1e-03	1.5e-02
1.0e-02	2.1e-02	1.1e-02	2.1e-02	3.1e-02	1.0e-02	3.2e-02
2.0e-02	4.4e-02	2.5e-02	4.5e-02	6.3e-02	2.2e-02	6.6e-02
5.0e-02	1.2e-01	8.3e-02	1.3e-01	1.7e-01	6.4e-02	1.9e-01
1.0e-01	3.1e-01	2.4e-01	3.5e-01	4.1e-01	1.9e-01	5.1e-01
2.0e-01	9.2e-01	8.6e-01	1.1e+00	1.3e+00	9.2e-01	1.9e+00

TABLE 1

VARIANCE OF THE EDGE JITTER VERSUS PROBABILITY OF POSITIVE IMPULSES.

n = 1			n = 2		
Std	Rec R	Rec F	Std	Rec R	Rec F
2.0e-05	1.5e-05	1.5e-05	3.0e-05	2.0e-05	2.0e-05
2.0e-04	1.5e-04	1.5e-04	3.0e-04	2.0e-04	2.0e-04
2.0e-03	1.5e-03	1.5e-03	3.0e-03	2.0e-03	2.0e-03
4.0e-03	3.0e-03	3.0e-03	6.0e-03	4.0e-03	4.0e-03
1.0e-02	7.6e-03	7.6e-03	1.5e-02	1.0e-02	1.0e-02
2.0e-02	1.6e-02	1.6e-02	3.0e-02	2.0e-02	2.0e-02
4.1e-02	3.2e-02	3.2e-02	5.9e-02	4.1e-02	4.1e-02
1.0e-01	8.9e-02	8.9e-02	1.4e-01	1.0e-01	1.0e-01
2.2e-01	2.1e-01	2.1e-01	2.8e-01	2.1e-01	2.1e-01
4.8e-01	5.4e-01	5.6e-01	5.7e-01	2.4e-01	4.2e-01

TABLE 2

VARIANCE OF THE EDGE JITTER VERSUS PROBABILITY OF POSITIVE AND NEGATIVE IMPULSES.

β = Probability of positive impulse; 2n + 1 = window length.
Rec : Recursive median filter - R : Rising edge - F : Falling edge.
Std : Standard median filter.

4. A CRITERIA FOR THE OPTIMALIZATION OF THE WINDOW LENGTH.

We have seen that the degree of signal smoothing increases as n increases; however, the edge jitter also increases as we increase the window size. Hence it would be desirable to find optimal window sizes to minimize signal distortion under some distortion criteria. The distortion measure should account for both, the edge jitter and the smoothing of the noise. Consider the distortion measure:

$$F(\gamma) = \gamma\, E\left\{ [\, x(i) - x_f(i)\,]^2 \right\} + (\, 1-\gamma\,)\, E\left\{ J^2 \right\} . \tag{34}$$

where x(i)=0 is the input signal before a rising edge and $x_f(i)$ is the output filtered signal; the expected value of J^2 (the jitter) can be determined with the help of (29b), (30c), (31) and (32). Minimizing the error measure with n as the varying parameter, we obtain the relation

$$(\, 1-\gamma\,)\, \beta \left[(\, n+1\,)^2 + 1 \right] = \gamma \left[1 + (n+1)\, \beta \right] . \tag{35}$$

Table 3 shows the values of n that should be chosen as a function of β and γ. From table 3 it can be observed that the more weight we put on edge jitter distortion, the smaller the window length must be; also, given a value γ, we must use longer windows for more smoothing. The results can be used for example in edge detection applications. Typically γ should be chosen small so that the weight of the edge jitter is increased.

5. CONCLUSION

A detailed analysis for the output distribution of a recursively i.i.d. median filtered sequence has been given. In particular, we derived the probability of positive impulses occurring at the output of the filter when the input is a constant plus impulsive noise. This for any window size. These results have been then compared to similar results obtained for standard filters and it is shown that the recursive median filter preforms better than the standard median filter in eliminating positive impulsive type noise. An extension of the analysis for white positive and negative impulsive type noise showed that the results are even better for the recursive median filter. The edge jitter introduced due to the filtering of a noisy edge has been then analyzed for white single sided and double sided impulsive type noise. Again the statistical description for the recursive and standard have been compared. We showed that the recursive median preserves better the original edges in a signal but generates a nonsymmetry between rising and falling edges. Finally a criteria for choosing an optimal window size is defined.

REFERENCES

[1] N. C. Gallagher and G. L. Wise, " *A Theorical Analysis of the Properties of Median Filters* " IEEE Trans. on ASSP, vol ASSP -29, no. 6, December 1981

[2] T. A. Nodes and N. C. Gallagher, " *Median Filters: Some Modifications and their Properties* " IEEE Trans. on ASSP, vol ASSP -30, no. 5, October 1982

[3] N. C. Gallagher, " *Median Filters: Properties and Application* " Presented at the 16th Asilomar Conference on Circuits, Systems, and Computers.

[4] G. R. Arce and N. C. Gallagher, " *State Description for the Root-Signals Set of Median Filters* " IEEE Trans. on ASSP, vol ASSP -30, no. 6, December 1982.

[5] G. R. Arce and N. C. Gallagher, " *Stochastic Analysis for the Multilevel Root-Signal Set of Median Filters* " Proceedings of the 16th annual Conference on Information Sciences and Systems. Princeton University, March 1982

[6] S. Karlin and H. Taylor, " *A First Course on Stochastic Processes* " Academic Press, sec. ed. 1975

[7] T. A. Nodes and N. C. Gallagher, " *Some Results on the Median Filtering of Signals and Additive White Noise* ", Proceedings of the 18th annual Allerton Conference on Communication, Control, and Computing. Urbana, October 1980

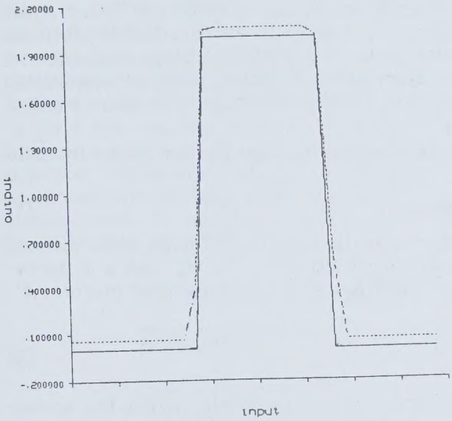


Fig 4a: For output y (line -) of a recursive median filter with an input signal plus positive impulsive noise of probability $\beta = 0.01$ the E { y } + $4\sigma_y$ (line ...)

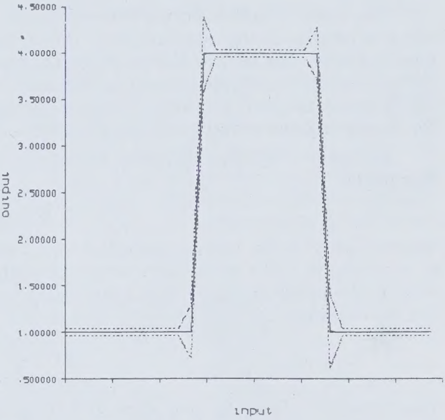


Fig 4b: For output y (line -) of a recursive median filter with an input signal plus positive and negative impulsive noise of probability $\beta = 0.01$ the E { y } + $4\sigma_y$ and the E { y } - $4\sigma_y$ (lower line ...)

β γ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
1.0e-03	10	15	20	25	31	38	48	64	98
2.0e-03	06	10	14	18	22	27	34	46	71
5.0e-03	04	06	08	11	14	17	22	29	46
1.0e-02	02	04	06	07	09	12	15	21	34
2.0e-02	01	03	04	05	07	08	11	15	25
5.0e-02	01	01	02	03	04	05	07	10	18
1.0e-01	01	01	01	02	03	04	05	08	14
2.0e-01	01	01	01	01	02	02	04	06	12

TABLE 3

Optimal value of n for a window length (2n+1) given the probability of positive impulses β and the weight $(1-\gamma)$ of the edge jitter.

SOLVING ILL-CONDITIONED SYSTEMS USING SINGULAR VALUE DECOMPOSITION WITH APPLICATION TO SIGNAL EXTRAPOLATION*

BARRY J. SULLIVAN and BEDE LIU
Department of Electrical Engineering and Computer Science
Princeton University
Princeton, New Jersey 08544

ABSTRACT

Singular value decomposition (SVD) is applied to solve the linear system $Ax=y$ where A is possibly non-square and ill-conditioned, and y may be corrupted by additive noise. An analysis of the error yields an expression that can be used to estimate the number of singular values needed to invert A. When the application is the extrapolation of discrete-time band-limited signals, examination of the error expression indicates that decimation can be used to reduce the computational load without significantly degrading the extrapolation.

SUMMARY

The need to solve an over-determined or under-determined system of linear equations arises in many applications. In addition, the matrix to be inverted is often ill-conditioned. Use of the Moore-Penrose inverse yields the minimum-norm least-squares (MNLS) solution, which is unique for any linear system. Singular value decomposition (SVD) provides both a means of implementing the Moore-Penrose inverse and a method for improving the stability of the problem.

Consider a system of linear equations with the observations contaminated by additive noise,

$$y = Ax + z, \tag{1}$$

where A is an $m \times n$ matrix operator with rank r, y is the m-vector of noisy observations, x is assumed to be a random n-vector with autocorrelation matrix R_x, and z is an m-vector of random noise with autocorrelation matrix R_z. The Moore-Penrose inverse, A^+, can be obtained through the SVD of A as [1]

$$A^+ = \sum_{i=1}^{r} \frac{1}{\lambda_i} \varphi_i \psi_i^*. \tag{2}$$

The singular values, λ_i, are arranged in decreasing order; the ψ_i and φ_i are the associated eigenvectors of AA^* and A^*A, respectively (* denotes conjugate transpose). When A is ill-conditioned, small singular values cause numerical instability, prompting the substitution of an l-truncated Moore-Penrose inverse,

$$A_l^+ \triangleq \sum_{i=1}^{l} \frac{1}{\lambda_i} \varphi_i \psi_i^* \quad (l \le r), \tag{3}$$

for A^+. The solution to (1) is then

$$\hat{x} \triangleq A_l^+ Ax + A_l^+ z. \tag{4}$$

An expression for the mean-square error in terms of the SVD of A and the Karhunen-Loéve transforms of x and z can be written as [2]

$$E(\|e\|^2) = E_t + E_n, \tag{5}$$

where $e \triangleq x - \hat{x}$, and

$$E_t \triangleq \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \sum_{j=1}^{l} \alpha_i |\varphi_j^* \xi_i|^2, \qquad E_n \triangleq \sum_{i=1}^{m} \sum_{j=1}^{l} \frac{\beta_i}{\lambda_j^2} |\psi_j^* \zeta_i|^2. \tag{6}$$

The scalers α_i and β_i and the vectors ξ_i and ζ_i are the eigenvalues and eigenvectors of R_x and R_z:

$$R_x \xi_i = \alpha_i \xi_i, \quad R_z \zeta_i = \beta_i \zeta_i. \tag{7}$$

---

*This work was supported by the Air Force Office of Scientific Research under contract AF-AFOSR-81-0186.

Note that E_l is decreasing in l and E_n is increasing in l. Equation (5) can be minimized through proper choice of l.

The extrapolation of a discrete-time band-limited signal can be written in the form of equation (1) by identifying A as the composite of the $m \times n$ truncation operator T with elements

$$t_{ij} = \begin{cases} 1 & i=j,\ |i| \le M \quad (i=0,\pm 1,\cdots,\pm M) \\ 0 & otherwise, \quad (j=0,\pm 1,\cdots,\pm N) \end{cases} \tag{8}$$

and the $n \times n$ band-limiting operator B with elements

$$b_{ij} = \frac{\sin 2\pi f_2(i-j)}{\pi(i-j)} - \frac{\sin 2\pi f_1(i-j)}{\pi(i-j)}, \tag{9}$$

where $m=2M+1$, $n=2N+1$, and $M \le N$. It is observed that for $A \triangleq TB$, the squares of the singular values are also the eigenvalues of the digital prolate spheroidal sequences (DPSS) [3,4]. Examination of the properties of these eigenvalues and equation (5) indicates that the sequence to be extrapolated need only be as long as the number of singular values used in forming A_l^+ [2]. In the case of an oversampled signal, decimation can be applied without significantly degrading the extrapolation. As the computational complexity of SVD is roughly $O(n^3)$, decimation by even a modest factor can result in substantial savings.

Examples of the application of SVD to signal extrapolation are summarized in the table below to illustrate these results. The estimate of the optimal number of singular values to use in forming the inverse (l) is seen to fall within two of the number observed to yield the smallest mean-square error (l_o). When compared to a recently proposed conjugate gradient extrapolation algorithm (CGEA) [5], the SVD approach is seen to be superior. The performance of the two approaches after decimation is also compared. As predicted, the extrapolation obtained via SVD shows little change in the observed mean-square error. In contrast, a substantial increase in the mean-square error is exhibited by CGEA after decimation.

TABLE

Rank and mean-square error for extrapolated signals

Method	m	n	f_1	f_2	SNR	l	l_o	Error (normalized)
SVD	45	195	0.0	0.02	90.0	8	6	0.3600
SVD	15	65	0.0	0.06	90.0	7	6	0.3560
SVD	9	39	0.0	0.1	90.0	7	6	0.3572
CGEA	45	195	0.0	0.02	90.0	-	-	0.6466
CGEA	15	65	0.0	0.06	90.0	-	-	0.6992
CGEA	9	39	0.0	0.1	90.0	-	-	0.7031

REFERENCES

[1] A. Ben-Israel and T. N. E. Greville, *Generalized Inverses: Theory and Applications*, New York: John Wiley and Sons, 1974.

[2] B. J. Sullivan and B. Liu, "On the use of singular value decomposition and decimation in discrete-time band-limited signal extrapolation," submitted to *IEEE Trans. on Acoust., Speech, and Signal Proc.*

[3] D. Slepian, "Prolate spheroidal wave functions, Fourier analysis, and uncertainty--V: The discrete case," *Bell Sys. Tech. J.*, vol. 57, no. 5, pp. 1371-1430, May-June 1978.

[4] A. D. Wyner, "An analog scrambling scheme which does not expand bandwidth, part I: Discrete time," *IEEE Trans. on Inform. Theory*, vol. IT-25, no. 3, pp. 261-274, May 1979.

[5] A. K. Jain and S. Ranganath, "Extrapolation algorithms for discrete signals with applications in spectral estimation," *IEEE Trans. on Acoust., Speech, and Signal Proc.*, vol. ASSP-29, no. 4, pp. 830-845, Aug. 1981.

APPROXIMATING THE EIGENVECTORS OF A SYMMETRIC TOEPLITZ MATRIX

DANIEL R. FUHRMANN and BEDE LIU
Department of Electrical Engineering and Computer Science
Princeton University
Princeton, New Jersey 08544

ABSTRACT

We present an algorithm, based on Wilkinson's eigenvalue/eigenvector perturbation formulae, for approximating the eigenvectors of a symmetric Toeplitz matrix. The computational complexity of the algorithm is $O(n^2 + mn\log n)$ where n is the order of the matrix and m is the desired number of eigenvectors. Simulation results indicate that the method is very powerful for locating the minimal eigenvector when the smallest eigenvalues are not closely spaced.

I. Introduction

At present there are a number of fast iterative algorithms for finding the minimum eigenvalue and associated eigenvector of a real symmetric Toeplitz matrix. Kung and Yu [1] have presented a couple of methods based on the inverse iteration method, and in a separate paper [2] we have presented a method based on the minimization of the Rayleigh quotient on the unit hypersphere. These methods have direct application to the 'sinusoids-in-noise' spectrum estimation problem, first posed by Pisarenko [3]. Other applications of the Toeplitz eigenvalue problem can be found in [4].

Any iterative eigenvalue algorithm will be greatly enhanced by a good first approximation to the eigenvectors. A good first approximation can save a large number of iterations with an algorithm such as Kung's; and in our algorithm [2], which can have poor initial convergence with a poor initial guess, such a 'pre-algorithm' is indispensable. In this paper, we present a one-pass method for approximating the lower eigenvectors, and as a by-product, we get reasonable approximations to all the eigenvalues, as well as upper and lower bounds on the smallest and largest eigenvalues, respectively. The algorithm requires $O(n^2)$ multiplications on a general-purpose processor, so the overhead in terms of computing time is reasonable. To achieve this low computational complexity, we require that n be a power of 2, although extensions of this method may be able to circumvent this requirement.

In our simulation runs with 32×32 matrices whose first-row elements were chosen at random, the vector which we identified as the approximate minimal eigenvector, formed a dot product with the true minimal eigenvector which was greater than .99 50% of the time and greater than .707 90% of the time. With additional refinement, we were able to obtain vectors whose dot product with the true minimal eigenvector was greater than .99 90% of the time. In about 7% of the cases, the chosen vector would be orthogonal to true minimal eigenvector; these were situations where the smallest eigenvalues were very close, and it was difficult to choose one vector out of several orthogonal vectors on the basis of a Rayleigh quotient test.

The algorithm is based on the notion of eigenvalue perturbations as set forth in Wilkinson [5]. The conceptual overview of the algorithm will be presented in section II, while most of the mathematical details will be developed in sections III through VI. A simple example is given in section VII. Section VIII contains the simulation results for 32×32

This work was supported by the Air Force Office of Scientific Research under contract AF-AFOSR-81-0186.

matrices and suggestions for further research.

II. Conceptual Overview of the Algorithm

Since the mathematical techniques to be employed in this paper are somewhat disparate, we will begin by presenting an overall picture of the algorithm.

Figure 1 contains what we will call an 'eigenvalue trajectory chart.' The points along the left axis correspond to the real eigenvalues of some symmetric matrix **A**. The points along the right axis represent the eigenvalues of the matrix **A** + **B**, and the curves between the two axes represent the loci of the eigenvalues of the matrix $\mathbf{C}(\varepsilon) = \mathbf{A} + \varepsilon\mathbf{B}$, as ε goes from 0 to 1.

In [5], Wilkinson gives formulae which can be interpreted as first and second derivatives of these eigenvalue trajectories at the points $\varepsilon = 0$ or $\varepsilon = 1$. Also given are formulae for the first order changes in the eigenvectors of **A**. These formulae require the eigenvectors of **A** and the elements of **B**, and in the general case, would be computationally prohibitive to implement ($O(n^3)$ multiplications). With certain classes of matrices **A** and **B**, however, the computational complexity can be reduced; in our case, we reduce it to $O(n^2)$ multiplications.

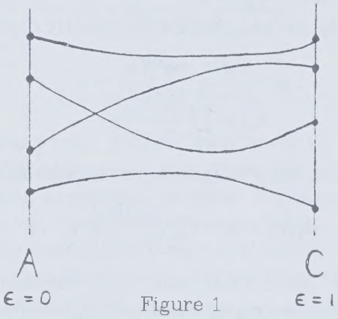


Figure 1

The outline of our algorithm is straightforward. Given a symmetric Toeplitz matrix **A**, we construct a symmetric matrix **C** = **A** + **B** which has three properties: 1) The eigenvectors of **C** are known, 2) the eigenvectors of **C** are close in some sense to the eigenvectors of **A**, and 3) **B** is small. Then using Wilkinson's formulae we calculate perturbations in the eigenvalues and eigenvectors of **C**, and use these as approximations to the eigenvalues and eigenvectors of **A**. Our approximation to the minimal eigenvector will most naturally be the one corresponding to the lowest approximate eigenvalue, although in some cases this is not always reliable as we shall see.

The above development gives a simple geometric interpretation of the algorithm, but the computational aspects are surprisingly different. In the next four sections, we will outline the mathematics which we will use to rapidly compute **C** and its perturbed eigenvectors.

III. Mathematical Preliminaries

For compactness, we introduce the following notation. Let **A**, **B**, and **C** be symmetric matrices with eigenvalues (α_i), (β_i), and (γ_i), respectively, and with orthonormal eigenvectors $(\mathbf{x}_i)$, $(\mathbf{y}_i)$, and $(\mathbf{z}_i)$, respectively. Define α as the vector of eigenvalues (α_i), and define β and γ similarly. Let $\tilde{\mathbf{A}} = diag\,(\alpha)$, $\tilde{\mathbf{B}} = diag\,(\beta)$, $\tilde{\mathbf{C}} = diag\,(\gamma)$. Let **X** be the modal matrix for **A**, whose i^{th} column is $\mathbf{x}_i$; define **Y** and **Z** similarly. It follows that $\mathbf{A} = \mathbf{X}\tilde{\mathbf{A}}\mathbf{X}^T$, $\mathbf{B} = \mathbf{Y}\tilde{\mathbf{B}}\mathbf{Y}^T$, and $\mathbf{C}=\mathbf{Z}\tilde{\mathbf{C}}\mathbf{Z}^T$. Finally, let **C** = **A** + **B**.

We define the Frobenius, or weak, norm of $\mathbf{B}$ as

$$|\mathbf{B}| = \left(\sum_i \sum_j b_{ij}^2\right)^{1/2}$$

This norm is invariant to orthogonal similarity transformations, hence

$$|\mathbf{B}| = \left(\sum_i \beta_i^2\right)^{1/2}$$

Our method is based on the theory of eigenvalue/eigenvector perturbations as set forth in Wilkinson [5]. It is a well-known known result of Ostrowski that the eigenvalues of a matrix are continuous with respect to changes in the elements of the matrix; this allows the construction of eigenvalue trajectories such as those in Fig. 1. If we consider the matrix $\mathbf{A}(\varepsilon) = \mathbf{C} - \varepsilon\mathbf{B}$, then the eigenvalues of $\mathbf{A}(\varepsilon)$ can be expressed as a convergent power series in ε:

$$\alpha_i(\varepsilon) = k_{i0} + k_{i1}\varepsilon + k_{i2}\varepsilon^2 \cdots$$

Clearly $k_{i0} = \alpha_i(0) = \gamma_i$, the i^{th} eigenvalue of $\mathbf{C}$.

Wilkinson's formulae for k_{i1} and k_{i2}, for the symmetric matrix case, are given by

$$k_{i1} = -\mathbf{z}_i^T \mathbf{B} \mathbf{z}_i \tag{3.1}$$

$$k_{i2} = \sum_{j \neq i} \frac{(\mathbf{z}_i^T \mathbf{B} \mathbf{z}_j)^2}{(\gamma_i - \gamma_j)} \tag{3.2}$$

The first-order approximation to the i^{th} eigenvector $\mathbf{x}_i$ is given by

$$\hat{\mathbf{x}}_i(\varepsilon) = \mathbf{z}_i - \varepsilon \sum_{j \neq i} \frac{(\mathbf{z}_i^T \mathbf{B} \mathbf{z}_j)}{(\gamma_i - \gamma_j)} \mathbf{z}_j \tag{3.3}$$

Simple examination of eqns. (3.1), (3.2), and (3.3) reveals that all the information required to compute the eigenvalue/eigenvector perturbations is contained in the matrix $\mathbf{X}^T\mathbf{B}\mathbf{X}$, and the vector of eigenvalues γ. Note that $(\mathbf{X}^T\mathbf{B}\mathbf{X})_{ij} = \mathbf{x}_i^T\mathbf{B}\mathbf{x}_j$.

It is our hope that, if $|\mathbf{B}| \ll |\mathbf{C}|$ then the above approximations will be reasonably valid even for $\varepsilon = 1$. If we construct a matrix $\mathbf{C}$ whose eigenvectors $\{\mathbf{z}_i\}$ are known, yet has the property that $|\mathbf{B}| = |\mathbf{C} - \mathbf{A}|$ is small compared to $\mathbf{C}$ or $\mathbf{A}$, then we can use these formulae to make educated guesses about the eigenstructure of $\mathbf{A}$. The theorem which follows gives us the method to form this construction.

THEOREM 1. For an arbitrary symmetric matrix $\mathbf{A}$, the matrix $\mathbf{C}$, which has a desired modal matrix $\mathbf{Z}$, and is chosen so that $|\mathbf{B}| = |\mathbf{C} - \mathbf{A}|$ is minimized, is given by $\mathbf{C} = \mathbf{Z}\tilde{\mathbf{C}}\mathbf{Z}^T$, where $\tilde{\mathbf{C}} = diag(\gamma)$, and $\gamma_i = \mathbf{z}_i^T\mathbf{A}\mathbf{z}_i$.

PROOF. Since the eigenvectors of $\mathbf{C}$ are fixed in advance, the only independent variables in the determination of $\mathbf{C}$ are its eigenvalues γ. We have from $\mathbf{C} = \mathbf{A} + \mathbf{B}$ that

$$\tilde{\mathbf{C}} = \mathbf{Z}^T\mathbf{A}\,\mathbf{Z} + \mathbf{Z}^T\mathbf{B}\,\mathbf{Z}$$

or

$$\mathbf{Z}^T\mathbf{B}\,\mathbf{Z} = \tilde{\mathbf{C}} - \mathbf{Z}^T\mathbf{A}\,\mathbf{Z}$$

Again, since $|\mathbf{B}| = |\mathbf{Z}^T\mathbf{B}\mathbf{Z}|$, we see it is sufficient to minimize the norm of $\tilde{\mathbf{C}} - \mathbf{Z}^T\mathbf{A}\mathbf{Z}$. Clearly this occurs when $\tilde{c}_{ii} = (\mathbf{Z}^T\mathbf{A}\mathbf{Z})_{ii} = \mathbf{z}_i^T\mathbf{A}\mathbf{z}_i$, since the only things we are at liberty to choose are the diagonal elements of $\tilde{\mathbf{C}}$. The off-diagonal elements of $\mathbf{Z}^T\mathbf{B}\mathbf{Z}$ remain constant, and so we see that $|\mathbf{B}|_{\min} = \left\{\sum_i \sum_{j \neq i} (\mathbf{Z}^T\mathbf{A}\mathbf{Z})_{ij}^2\right\}^{1/2}$.

One interesting property of $\mathbf{Z}^T\mathbf{BZ}$ is that its diagonal is all-zero. This leads to *trace* $(\mathbf{B}) = 0$, hence $\sum_i \gamma_i = \sum_i \alpha_i$. It can also be shown that each eigenvalue γ_i is a convex combination of the α's. This leads to

$$\gamma_{\min} \geq \alpha_{\min} \text{ and } \gamma_{\max} \leq \alpha_{\max}$$

This can be deduced as well from the fact that γ_i is the quadratic form $\mathbf{x}^T\mathbf{Ax}$ evaluated at the unit vector $\mathbf{z}_i$. This quadratic form achieves its minimum and maximum (over unit vectors) at the eigenvectors corresponding to the minimum and maximum eigenvalues of $\mathbf{A}$, respectively.

The primary implication of theorem 1 is one of implementation, however. Recall from eqns. (3.1) - (3.3) that we seek the elements of $\mathbf{Z}^T\mathbf{BZ}$. If $\mathbf{B}$ and $\mathbf{C}$ are constructed according to theorem 1, then $\mathbf{z}_i^T\mathbf{B}\mathbf{z}_i = 0$, and $\mathbf{z}_i^T\mathbf{B}\mathbf{z}_j = -(\mathbf{Z}^T\mathbf{AZ})_{ij}$. In addition, $\gamma_i = (\mathbf{Z}^T\mathbf{AZ})_{ii}$. This tells us that, in terms of computation, the matrices $\mathbf{B}$ and $\mathbf{C}$ need never be numerically constructed: all the necessary information is in the matrix $\mathbf{Z}^T\mathbf{AZ}$. If we let $\hat{\mathbf{A}} = \mathbf{Z}^T\mathbf{AZ}$, then the approximation equations become

$$\hat{\alpha}_i = \hat{a}_{ii} + \sum_{j \neq i} \frac{\hat{a}_{ij}^2}{\hat{a}_{ii} - \hat{a}_{jj}} \tag{3.4}$$

$$\hat{\mathbf{x}}_i = \mathbf{z}_i + \sum_{j \neq i} \frac{\hat{a}_{ij}}{\hat{a}_{ii} - \hat{a}_{jj}} \mathbf{z}_j \tag{3.5}$$

We have stated previously that, given $\mathbf{A}$, we hope to be able to construct a matrix $\mathbf{C}$ such that $\mathbf{B}$ is small and the above approximations will be valid. Theorem 1 gave us a method for making $\mathbf{B}$ as small as possible, however, it gives us no indication of whether or not $\mathbf{B}$ is small enough. The last result of this section tells us that the size of $\mathbf{B}$ depends on the relationship between the fixed eigenvectors of $\mathbf{C}$ and the true (unknown) eigenvectors of $\mathbf{A}$. For example, if $\mathbf{Z} = \mathbf{X}$, then $\mathbf{B} = 0$, since $\mathbf{A}$ already has the desired eigenvectors.

Let $\psi_i = \sin(\mathbf{z}_i, \mathbf{x}_i) = \sqrt{1 - (\mathbf{z}_i^T\mathbf{x}_i)^2}$, where $\mathbf{z}_i$ and $\mathbf{x}_i$ are the i^{th} column of $\mathbf{Z}$ and $\mathbf{X}$, respectively. As the columns of $\mathbf{Z}$ become more and more closely aligned with those of $\mathbf{X}$, the ψ_i's go to zero. We define $\psi = \psi_{\max}$ to be the measure of coincidence between $\mathbf{Z}$ and $\mathbf{X}$.

THEOREM 2. Given an arbitrary $\mathbf{A}$, and a modal matrix $\mathbf{Z}$, choose $\mathbf{B}$ according to Theorem 1. Define ψ as above; $\max_i(\sin(\mathbf{z}_i, \mathbf{x}_i))$. Then $|\mathbf{B}| \leq t\psi^2$, where t is a fixed function of $\mathbf{A}$.

PROOF. Let $\mathbf{F} = \mathbf{Z}^T\mathbf{X}$ $f_{ii} = \mathbf{z}_i^T\mathbf{x}_i = \sqrt{1 - \psi_i^2}$. Since $\mathbf{F}$ is an orthogonal matrix, $\sum_{j \neq i} f_{ij}^2 = \psi_i^2$. Also, $|f_{ij}| \leq \psi_i$, for $i \neq j$.

From theorem 1, we have $|\mathbf{B}| = (\sum_i \sum_{j \neq i} (\mathbf{Z}^T\mathbf{AZ})_{ij}^2)^{1/2}$. We know as well that $\mathbf{Z}^T\mathbf{AZ} = \mathbf{F}^T\tilde{\mathbf{A}}\mathbf{F}$.

Now

$$(\mathbf{F}^T\tilde{\mathbf{A}}\mathbf{F})_{ij} = \sum_k f_{ki} f_{kj} \alpha_k$$

$$\left|\sum_k f_{ki} f_{kj} \alpha_k\right| \leq \sum_k |f_{ki} f_{kj} \alpha_k|$$

$$= \sum_k |f_{ki}||f_{kj}||\alpha_k|$$

Since $|f_{ki}| \leq \psi_k \leq \psi$, we have $|(\mathbf{F}^T\tilde{\mathbf{A}}\mathbf{F})_{ij}| \leq \psi^2 \sum_k |\alpha_k|$

Therefore $|\mathbf{B}| \leq \psi^2 \sqrt{(n^2 - n)(\sum_k |\alpha_k|)^2}$. The quantity under the radical is the constant t which is mentioned in the statement of the theorem.

IV. Choosing the Modal Matrix Z

In this section, we shall at last make use of the fact that **A** is a symmetric Toeplitz matrix. Notice that the development of section III applies to any symmetric or Hermitian matrix. We can see, however, that the pivotal operations of the algorithm are the choice of **Z** and the fast computation of $\mathbf{Z}^T\mathbf{AZ}$. For an arbitrary **A**, we have no clue as to what set of vectors **Z** may be close to the true eigenvectors **X**; furthermore, the computation of $\mathbf{Z}^T\mathbf{AZ}$ is an $O(n^3)$ operation. By restricting **A** to be Toeplitz, we can exploit some of the known properties of eigenvectors of Toeplitz matrices [6], and at the same time, arrive at a **Z** such that $\mathbf{Z}^T\mathbf{AZ}$ can be computed in $O(n^2)$ time.

Let us briefly review some of these properties. First, half of the eigenvectors are reciprocal and half are anti-reciprocal. This means that for half of the vectors, $\mathbf{z}_i(k) = \mathbf{z}_i(n-1-k)$, and for the other half, $\mathbf{z}_i(k) = -\mathbf{z}_i(n-1-k)$.* This is a property shared by all matrices which are symmetric about the main and the secondary diagonal. We should mention that, even if **A** has multiple eigenvalues and there are an infinite number of eigenvectors, an orthonormal set of eigenvectors can always be chosen that has this property.

The second interesting property of Toeplitz eigenvectors is that the eigenfilter, or polynomial, associated with the maximal and minimal eigenvectors always has its zeroes on the unit circle. The eigenfilter is a polynomial over the complex field whose coefficients are given by the elements of the associated eigenvector. Again in the case of multiple minimum eigenvalues, we can always choose a eigenvector from the minimal (maximal) eigenspace which has this property.

A third, and most useful, property of Toeplitz eigenvectors is that there exists a whole class of Toeplitz matrices which has a constant set of eigenvectors. It is exactly these constant vectors which we shall use to form **Z**. The class we refer to is, of course, the symmetric circulant matrices, whose properties are spelled out by Gray in [7].

A circulant matrix is one in which each row is a circular shift (to the right) of the one above it. A symmetric circulant matrix is a Toeplitz matrix whose first row **c** has the additional property that $c_i = c_{n-i}$. As Gray points out, the eigenvectors of **C** are the columns of the DFT (Discrete Fourier Transform) matrix **W**, and its eigenvalues γ_i are just the DFT of the first row or column of **C**. Since $c_i = c_{n-i}$, we have from the properties of the DFT that $\gamma_i = \gamma_{n-i}$, and that the γ's are all real (which we knew anyway).

Let us denote the k^{th} column of **W** as $\mathbf{w}_k$. Recall that

$$\mathbf{w}_k(i) = W_n^{ik}, \tag{4.1}$$

where $W_n = \exp(-j2\pi/n)$. Since γ_k and γ_{n-k} form a double-eigenvalue pair, any linear combination of $\mathbf{w}_k$ and $\mathbf{w}_{n-k}$ is also an eigenvector of **C**. For each $(\mathbf{w}_k, \mathbf{w}_{n-k})$ pair, we will construct two orthogonal vectors which satisfy all the above properties of Toeplitz eigenvectors. We shall denote these vectors with the subscripts r and ar, for reciprocal and anti-reciprocal, respectively.

$$\mathbf{z}_{k,r} = \mathbf{w}_k \; W_n^{k/2} + \mathbf{w}_{n-k} \; W_n^{-k/2} \tag{4.2}$$

$$\mathbf{z}_{k,ar} = j\,\mathbf{w}_k \; W_n^{k/2} - j\,\mathbf{w}_{n-k} \; W_n^{-k/2}, \quad 1 \leq k \leq \frac{n}{2} - 1 \tag{4.3}$$

---

* We use the indexing convention that $0 \leq k \leq n-1$.

Substituting eqn. (4.1), we see that

$$\begin{aligned} \mathbf{z}_{k,r}(i) &= W_n^{ik}\, W_n^{\frac{1}{2}k} + W_n^{-ik}\, W_n^{-\frac{1}{2}k} \\ &= W_n^{(i+\frac{1}{2})k} + W_n^{-(i+\frac{1}{2})k} = 2\cos\left[\frac{2\pi k}{n}(i+\tfrac{1}{2})\right] \end{aligned} \tag{4.4}$$

$$\begin{aligned} \mathbf{z}_{k,ar}(i) &= jW_n^{ik}\, W_n^{\frac{1}{2}k} - jW_n^{-ik}\, W_n^{-\frac{1}{2}k} \\ &= jW_n^{(i+\frac{1}{2})k} - jW_n^{-(i+\frac{1}{2})k} = 2\sin\left[\frac{2\pi k}{n}(i+\tfrac{1}{2})\right] \end{aligned} \tag{4.5}$$

In addition, we choose $\mathbf{z}_0 = \mathbf{w}_0$ and $\mathbf{z}_{n/2} = \mathbf{w}_{n/2}$. We can now define our modal matrix $\mathbf{Z}$ by letting the left $n/2$ columns be reciprocal and the right half be anti-reciprocal. That is to say, let $\mathbf{z}_k = \mathbf{z}_{k,r}$ and $\mathbf{z}_{n-k} = \mathbf{z}_{k,ar}$ for $1 \le k \le \frac{n}{2} - 1$. Of course, we must not forget to orthonormalize $\mathbf{Z}$; toward this end we multiply $\mathbf{z}_0$ and $\mathbf{z}_{n/2}$ by $\frac{1}{\sqrt{n}}$ and the rest by $\frac{1}{\sqrt{2n}}$. To summarize, we can simply define $\mathbf{Z}$ in shorthand notation by eqn. (4.6).

$$\mathbf{Z} = \frac{1}{\sqrt{2n}}\mathbf{W}\mathbf{K} \tag{4.6}$$

$$\text{where } \mathbf{K} = \begin{bmatrix} \sqrt{2} & 0 & \cdot & \cdot & \cdot & \cdot & 0 \\ 0 & W_n^{\frac{1}{2}} & & & & & jW_n^{\frac{1}{2}} \\ \cdot & \cdot & W_n^{1} & & & \cdot & \\ \cdot & & & \cdot & \cdot & & \\ \cdot & & & & \sqrt{2} & & \\ \cdot & & & \cdot & \cdot & & \\ \cdot & & W_n^{-1} & & & \cdot & \\ 0 & W_n^{-\frac{1}{2}} & & & & & -jW_n^{-\frac{1}{2}} \end{bmatrix}$$

The reader can quickly verify that the columns of $\mathbf{Z}$ satisfy the properties given earlier. In particular, the eigenfilter property is satisfied for the minimal and maximal eigenvectors since *all* of the eigenfilters associated with $\mathbf{Z}$ have their zeroes on the unit circle.

V. Fast Computation of $\mathbf{Z}^T \mathbf{A}\, \mathbf{Z}$

Our experience has shown that the columns of $\mathbf{Z}$ form a fair approximation to the eigenvectors of $\mathbf{A}$ in many cases. This helps to ensure that the matrix $\mathbf{B}$ is small and that further refinements of the approximate eigenvectors will be accurate, as we saw in section 3. This particular choice of $\mathbf{Z}$ has an additional advantage in terms of computation as well, and in fact it is possible to compute $\mathbf{Z}^T\mathbf{A}\mathbf{Z}$ with $O(n^2)$ multiplications.

Referring back to eqn. (4.6), we can write

$\mathbf{Z}^T\mathbf{A}\mathbf{Z} = \frac{1}{2n}\mathbf{K}^T\, \mathbf{W}^T\mathbf{A}\mathbf{W}\mathbf{K}$, and *since* $\mathbf{W}$ *is symmetric*,

$$\mathbf{Z}^T\mathbf{A}\mathbf{Z} = \frac{1}{2n}\mathbf{K}^T\mathbf{W}\mathbf{A}\mathbf{W}\mathbf{K} \tag{5.1}$$

We recognize right away that $\mathbf{WAW}$ is the 2-D DFT of $\mathbf{A}$. This operation can be performed in $O(n^2 \log n)$ multiplication for an arbitrary matrix $\mathbf{A}$, just by doing row-column FFT operations or by using the simultaneous decimation algorithm of [8]. We can exploit the Toeplitz nature of $\mathbf{A}$ to achieve an even further savings, however, by making a simple modification to the simultaneous decimation algorithm. The algorithm which we have developed requires only $O(n^2)$ multiplications, provided n is a power of 2. The details of the Toeplitz 2-D FFT will not be discussed here, however. See [9].

Once we have computed $\hat{\mathbf{A}} = \mathbf{WAW}$, the computation of $\mathbf{K}^T\hat{\mathbf{A}}\mathbf{K}$ is fairly straightforward. From the sparse nature of $\mathbf{K}$, this is also an $O(n^2)$ operation. We refer the reader to [9] for the details of this operation as well.

One of the properties of $\mathbf{Z}^T\mathbf{AZ}$ is that its upper-right and lower-left quadrants are all-zero. This is a result of our placing the reciprocal vectors in the left half of $\mathbf{Z}$, and the anti-reciprocal vectors in the right half. Since $(\mathbf{Z}^T\mathbf{AZ})_{ij} = \mathbf{z}_i^T\mathbf{Az}_j$, we have for $i \geq \frac{n}{2}$ and $j < \frac{n}{2}$, that $\mathbf{Az}_j$ is a reciprocal vector, and the inner product of $\mathbf{z}_i$ and $\mathbf{Az}_j$ is therefore 0. A similar argument applies when $i < \frac{n}{2}$ and $j \geq \frac{n}{2}$.

VI. Computation of Perturbed Eigenvalues & Eigenvectors

Once $\hat{\mathbf{A}} = \mathbf{Z}^T\mathbf{AZ}$ has been computed by our quick method, the only remaining task is to implement eqns. (3.4) and (3.5) to arrive at the values of $\hat{\alpha}_i$ and $\hat{\mathbf{x}}_i$, which will serve as our estimates of α_i and $\mathbf{x}_i$. We will assume that we examine the quantities $\hat{a}_{ii}$ to determine which eigenvectors and eigenvalues we wish to compute. If, for instance, we want to find k lower eigenvectors, we choose $i_1, i_2, \cdots i_k$, corresponding to the k lowest values of $\hat{a}_{ii}$. Since the ordering of $\hat{\alpha}_i$'s may not be the same as the ordering of $\hat{a}_{ii}$, we will normally examine a few more than k values of $\hat{\alpha}_i$ and $\hat{\mathbf{x}}_i$ in order to find the k eigenvectors which approximate the k lowest.

Computation of $\hat{\alpha}_i$ is straightforward according to eqn. (3.4) : $2(n-1)$ multiplications are required. If we temporarily save the quantities $\left\{\frac{\hat{a}_{ij}}{\hat{a}_{ii} - \hat{a}_{jj}}\right\}$ when computing $\hat{\alpha}_i$, they can be used in the computation of $\hat{\mathbf{x}}_i$.

We see from eqn. (3.5) that $\hat{\mathbf{x}}_i$ is a linear combination of the $\mathbf{z}_i$; we can rewrite eqn. (3.5) as

$$\hat{\mathbf{x}}_i = \mathbf{Zb}_i \tag{6.1}$$

$$\text{where} \quad \mathbf{b}_i(j) = \frac{\hat{a}_{ij}}{(\hat{a}_{ii} - \hat{a}_{jj})}, \qquad j \neq i \tag{6.2}$$

$$\mathbf{b}_i(i) = 1 \tag{6.3}$$

Recalling that $\mathbf{Z} = \frac{1}{\sqrt{2n}}\mathbf{WK}$, we find that

$$\hat{\mathbf{x}}_i = \frac{1}{\sqrt{2n}}\mathbf{WKb}_i \tag{6.4}$$

Again, by the sparse nature of $\mathbf{K}$, $\mathbf{Kb}$ is computed in $O(n)$ time. Additional savings are also possible from the fact that $\mathbf{b}_i(j) = 0$ for $j \geq \frac{n}{2}$ if $i < \frac{n}{2}$, or alternatively $\mathbf{b}_i(j) = 0$ if $j < \frac{n}{2}$ and $i \geq \frac{n}{2}$. With the exception of a multiplicative constant, then, $\hat{\mathbf{x}}_i = DFT(\mathbf{Kb}_i)$.

This operation can be performed by an appropriate FFT routine in $O(n \log n)$ time. If we desire a normalized version of $\hat{\mathbf{x}}_i$, then we can forget about the $\frac{1}{\sqrt{2n}}$-factor and just normalize $\mathbf{WKb}_i$. This is usually the best procedure since $\mathbf{b}_i$ can, in general, have arbitrary norm.

We see from this analysis that the limiting factor in computing $\hat{\alpha}_i$ and $\hat{\mathbf{x}}_i$ is the computation of $\mathbf{WKb}_i$. We conclude then that, in addition to $O(n^2)$ multiplications to find $\mathbf{Z}^T\mathbf{AZ}$, we perform $O(mn \log n)$ multiplications to find m approximate eigenvalues and eigenvectors.

VII. Example

At this point, we illustrate the computational aspects of our algorithm with a small example.

$$\text{Let } \mathbf{A} = \begin{bmatrix} 5 & 4 & 0 & -2 \\ 4 & 5 & 4 & 0 \\ 0 & 4 & 5 & 4 \\ -2 & 0 & 4 & 5 \end{bmatrix}$$

The true eigenvalues of $\mathbf{A}$ are (11, 9, 1, -1) and the corresponding eigenvectors, given in matrix form, are

$$\mathbf{X} = \frac{1}{\sqrt{10}} \begin{bmatrix} 2 & 1 & 2 & 1 \\ 1 & -2 & 1 & -2 \\ 1 & -2 & -1 & 2 \\ 2 & 1 & -2 & -1 \end{bmatrix}$$

The 4×4 $\mathbf{Z}$ matrix is

$$\mathbf{Z} = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \end{bmatrix}$$

$$\mathbf{Z}^T\mathbf{AZ} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} 5 & 4 & 0 & -2 \\ 4 & 5 & 4 & 0 \\ 0 & 4 & 5 & 0 \\ -2 & 0 & 4 & 5 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \end{bmatrix}$$

$$= \begin{bmatrix} 10 & -3 & 0 & 0 \\ -3 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 \\ 0 & 0 & 3 & 8 \end{bmatrix}$$

The corresponding $\mathbf{C}$ and $\mathbf{B}$ from theorem 1 are

$$\mathbf{C} = \begin{bmatrix} 5 & 4 & 0 & 1 \\ 4 & 5 & 1 & 0 \\ 0 & 1 & 5 & 4 \\ 1 & 0 & 4 & 5 \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 3 \\ 0 & 0 & -3 & 0 \\ 0 & -3 & 0 & 0 \\ 3 & 0 & 0 & 0 \end{bmatrix}$$

Our approximations $\hat{\alpha}_i$ and $\hat{\mathbf{x}}_i$ are given by

$$\hat{\alpha}_1 = 10 + \frac{3^2}{(10-2)} = 11.125$$

$$\hat{\alpha}_2 = 2 + \frac{3^2}{(2-10)} = .825$$

$$\hat{\alpha}_3 = 0 - \frac{3^2}{8} = -1.125$$

$$\hat{\alpha}_4 = 8 + \frac{3^2}{8} = 9.125$$

$$\hat{\mathbf{x}}_1 = \mathbf{Z}\begin{bmatrix} 1 \\ \frac{3}{8} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 11/8 \\ 5/8 \\ 5/8 \\ 11/8 \end{bmatrix} = \begin{bmatrix} 11 \\ 5 \\ 5 \\ 11 \end{bmatrix} = \begin{bmatrix} 2.2 \\ 1.0 \\ 1.0 \\ 2.2 \end{bmatrix} \frac{1}{\sqrt{11.68}}$$

$$\hat{\mathbf{x}}_3 = \mathbf{Z}\begin{bmatrix} 0 \\ 0 \\ 1 \\ -\frac{3}{8} \end{bmatrix} = \begin{bmatrix} 5/8 \\ -11/8 \\ +11/8 \\ 5/8 \end{bmatrix} = \begin{bmatrix} 5 \\ -11 \\ 11 \\ -5 \end{bmatrix} = \begin{bmatrix} 1.0 \\ -2.2 \\ 2.2 \\ -1.0 \end{bmatrix} \frac{1}{\sqrt{11.68}}$$

Comparison of $\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_3$ with $\mathbf{x}_1$ and $\mathbf{x}_3$ shows that $\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_3$ would serve as good first approximations to $\mathbf{x}_1$ and $\mathbf{x}_3$. The dot product $\hat{\mathbf{x}}_1^T\mathbf{x}_1$ and $\hat{\mathbf{x}}_3^T\mathbf{x}_3$ is .999.

VIII. Simulation Results

We applied the algorithm to a number of 32 × 32 symmetric Toeplitz matrices. The elements of the first row of each matrix were generated by a Gaussian pseudo-random number generator.

The algorithm was used to compute 4 approximate eigenvectors, corresponding to the 4 lowest values of a_{ii}. The four values of α_i were also computed, but the simulation showed that, while the $\hat{\alpha}_i$ provide a good approximation to the distribution of true eigenvalues α_i, they were not reliable enough for us to be able to choose an approximate minimum vector from the four candidates. We therefore resorted to using a Rayleigh quotient test to choose the minimum, i.e., we choose $\hat{\mathbf{x}}_{\min}$ to be that $\hat{\mathbf{x}}_i$ which minimized the Rayleigh quotient $\mathbf{x}_i^T\mathbf{A}\mathbf{x}_i$. Fortunately, the procedure for doing this is of $O(n \log n)$ complexity, due to the Toeplitz nature of A. The procedure for doing this is described in another paper [2].

We used as a performance measure the inner product p of our approximate minimal vector with the true minimal eigenvector calculated according to the IMSL subroutine EIGRS. Out of 200 trials calculated, 50% of the time the algorithm returned vectors which $p > .99$, and in 90% of the cases, we had $p > .707$. When the true lower eigenvalues were closely spaced, however, the algorithm had difficulty in selecting the correct minimum vector, since the Rayleigh quotients $\mathbf{x}_i^T\mathbf{A}\mathbf{x}_i$ were very nearly the same. As a result, in 10% of our trials, we got $p = 0$.

IX. Conclusion

The algorithm we have presented for finding eigenvectors of symmetric Toeplitz matrices is fast and reasonably accurate. If it is to be used in calculation of the minimal eigenvector, however, it cannot stand alone. The returned vectors need to be refined somewhat before a decision can be made about which candidate vector is close to the true minimal eigenvector.

We intend to do further research along these lines. In particular, this algorithm can be used in conjunction with some iterative eigenvalue routine such as the Rayleigh Quotient Iteration [1] or our Constrained Conjugate Gradient method [2]. Preliminary indications are that we can very quickly (in $O(n \log n)$ time) produce vectors for which $p > .99$ 90% of the time, with a 7% failure rate.

References

1. Y. H. Hu and S. Y. Kung, "Highly Concurrent Toeplitz Eigensystem Solver for High Resolution Spectral Estimation", *Proc. ICASSP 83*, pp. 1422-1425, April 1983.
2. D. Fuhrmann and B. Liu, "An Iterative Algorithm for Finding the Minimum Eigenvalue of a Class of Symmetric Matrices", submitted to *ICASSP 84*.
3. V. F. Pisarenko, "The Retrieval of Harmonics from a Covariance Function", *Geophysical J. Royal Astronomical Soc.*, Vol. 33, pp. 347-366, 1973.
4. U. Grenander and G. Szego, *Toeplitz Forms and Their Applications*, University of California Press, Berkeley, CA, 1958.
5. J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Oxford University Press, Oxford, 1965.
6. J. Makhoul, "On the Eigenvectors of Symmetric Toeplitz Matrices", *IEEE Trans. on ASSP*, vol. ASSP-29, no. 4, August 1981.
7. R. M. Gray, *Toeplitz and Circulant Matrices: II*, Stanford University Information Systems Laboratory Technical Report No. 6504-1.
8. E. Hoyer and W. Berry, "An Algorithm for the Two Dimensional FFT", *Proc. 1977 ICASSP*, Hartford, Conn., pp. 552-555.
9. D. Fuhrmann and B. Liu, "Approximating the Eigenvectors of a Symmetric Toeplitz Matrix", to be published.

INDEX OF AUTHORS

Aggarwal, A. 236,258
Ahmad, H. K. 190
Allenbach, J. P. 110
Apostolico, A. 70
Arazi, B. 520
Arce, G. R. 1034
Aşkar, M. 613
Bajaj, C. 459
Balasubramanian, R. 23,760
Bars, R. 693
Bauer, R. D. 152
Bavarian, B. 312
Bayoumi, M. 1003
Belbas, S. A. 734
Bentley, J. L. 51
Benvenuto, N. 993
Berger, T. 293
Berman, F. 214
Bhargava, V. K. 511
Bhattacharyya, S. P. 744,758
Bingulac, S. P. 393,423,920
Bose, B. 812
Bozzo, C. 425
Bredeson, J. G. 806
Brower, A. S. 403
Burns, M. 754
Calović, M. 896
Calvet, J. L. 343
Capocelli, R. M. 528
Carlson, D. A. 80
Carraway, P. I. 911
Chang, C. C. 675
Chang, R. C. 461
Chazelle, B. 427,447
Chen, S. 732
Chen, W.-C. 683
Chern, M.-Y. 863
Chiasson, J. 21
Chyung, D. H. 752
Cioffi, G. 852
Cobb, J. D. 908
Cohoon, J. 246
Cooper, E. H. 971
Cooper, G. R. 971
Coşkunoǧlu, O. 770
Coyle, E. J. 170,1024
Crinon, R. J. 1034
Čualevski, N. 896
Dandapani, R. 796
Davies, O. J. 822
Delp, E. J. 132,142
Derin, H. 613
Diduch, C. P. 760
Dungern, O. V. 832
Dudek, K. P. 25
Eichel, P. H. 142
Ellis, J. A. 224
Emre, E. 11,564

Erickson, K. T. 983
Ericson, T. 277
Ficola, A. 335
Fischer, T. R. 501
Fitch, J. P. 1024
Fleming, J. A. 758
Flentz, J. M. 712
Fouques, M. 425
Franks, L. E. 993
Frederickson, G. N. 637
Friesen, D. K. 61
Fuhrmann, D. R. 1046
Gajda, W. J., Jr. 712
Gallagher, N. C., Jr. 1024
Gertler, J. 722
Gluhajic, N. 413
Guerrieri, E. 70
Hakimi, S. L. 353
Halverson, D. R. 622
Hambrusch, S. E. 256
Havlicsek, B. L. 363
Heegard, C. 293
Hill, F. S., Jr. 993
Howze, J. W. 744
Huang, K.-T. 842
Huang, Y. F. 712
Hwang, I. S. 383
Incerpi, J. 447
Ioannou, P. A. 703
Ip, A. 832
Ismail, M. 774
Javid, S. H. 403
Johnson, D. S. 51
Jullien, G. A. 1003
Jun, W. J. 964
Kasami, T. 866
Kasavin, A. D. 202
Kashyap, R. L. 152
Kassam, S. A. 1014
Keel, L. H. 758
Kelly, P. A. 602
Keviczky, L. 693
Kheradpir, S. 303
Kim, H. K. 774
Kim, J. 752
Kime, C. R. 383
Knoll, T. F. 132
Kreutzer, S. E. 353
Krogh, B. H. 325
Krothapalli, V. P. 373
Kumar, K. S. P. 693
Lai, T.-H. 886
Langston, M. A. 61
Lazar, A. A. 170,180,190,200
Lea, C.-T. A. 160
Lee, C. C. 90
Lee, D. T. 90
Lee, R. C. T. 461
Lee, Y. H. 1014
Leighton, F. T. 51

Leininger, G. G. 722
Leiserson, C. E. 204
Lenhart, S. M. 734
Leung, J. Y.-T. 876
Li, M. 459
Liebetreu, J. M. 972
Lin, D. J. 812
Lindner, D. K. 554
Liu, B. 1044,1046
Lodaya, M. D. 772
Lombardi, F. 882
Lu, F. S. 481
Lucertini, M. 852
Maki, G. K. 786
Malek, M. 776
Marino, R. 335
Martinez, A. B. 631
Massey, J. L. 268
Mayes, L. 981
McIntyre, M. 35
McGeoch, C. C. 51
Medanić, J. 413,896
Meditch, J. S. 160
van der Meulen 284
Meyer, G. G. L. 345,363,610
Michel, A. N. 983
Mielke, R. R. 911
Miller, W. C. 1003
Mintz, M. 592
Mitrović, V. 920
Moayeri, N. 120
Morgan, R. 315
Motohashi, Y. 866
Murata, T. 863
Myers, J. R. 202
Nakajima, K. 373
Natarajan, K. 23
Neuhoff, D. L. 120
Nicosia, S. 335
Ohm, D. 744
Okamoto, T. 866
Opdenacker, Ph. 703
Opper, E. 776
O'Reilly, J. J. 964
Ozguner, F. 832
Özgüner, Ü. 315,754
Papavassilopoulos, G. P. 930
Petranović, D. 413
Naraghi-Pour, M. 120
Radhakrishnan, D. 786
Reid, R. E. 770
Robertazzi, T. G. 180
Roda, V. O. 822
Root, W. L. 1
Sack, J.-R. 437
Saha, D. 954
Sahni, S. 246,886
Sain, M. K. 25
Saluja, K. K. 796
Sandberg, I. W. 546

Sawan, M. E. 933
Saxe, J. B. 204
Schalkwijk, J. P. M. 284
Benyamin-Seeyar, A. 511
Sennott, J. W. 130
Shayman, M. A. 582
Shiva, S. G. S. 511
Siffredi, E. 425
Singh, H. 806
Sinha, A. S. C. 573
Ilic-Spong, M. 935
Spong, M. W. 43,303
Standley, H. M. 234
Su, D. H. 675
Sudborough, I. H. 224
Sullivan, B. J. 1044
Suzuki, I. 866
Swaszek, P. F. 491
Taniguchi, K. 866
Thorp, J. S. 303
Toussaint, G. T. 457,471
Tran, M. T. 933
Tuǧcu, A. K. 770
Turner, J. S. 224
Tu, M. 930
Ursic, S. 100
Vaccaro, U. 528
Vaishnavi, V. K. 647
Vakil, F. 200
Varghese, M. 935
Vidyasagar, M. 35
Vitter, J. S. 683
Ward, R. 537
Weinert, H. L. 610
West, P. J. 423
Wickert, M. A. 945
Willard, D. E. 656,663
Willems, F. M. J. 284
Winkelman, J. R. 403
Wise, G. L. 481,622
Wong, C. K. 90
Wu, C. 722
Wyman, B. F. 25
Yedavalli, R. K. 756
Zeytinoǧlu, M. 592
Ziemer, R. E. 945,972

UNIVERSITY OF ILLINOIS-URBANA
621.3811AL5 C001
PROCEEDINGS / ALLERTON CONFERENCE ON COM
1983
3 0112 008149186